IMPACT: Importance-Aware Activation Space Reconstruction

Md Mokarram Chowdhury[1]*, **Daniel Agyei Asante**[1], **Ernie Chang**[2], **Yang Li**[1]*†
[1]Department of Computer Science, Iowa State University, United States
[2]Meta, United States
{mokarram, dasante, yangli1}@iastate.edu, erniecyc@meta.com

Abstract

Large language models (LLMs) achieve strong performance across diverse domains but remain difficult to deploy in resource-constrained environments due to their size. Low-rank compression is a common remedy, typically minimizing weight reconstruction error under the assumption that weights are low-rank. However, this assumption often does not hold in LLMs. In contrast, LLM activations exhibit a more pronounced low-rank structure, motivating approaches that minimize activation reconstruction error.

This shift alone, however, is not sufficient: different activation dimensions contribute unequally to model performance, and treating them uniformly can lead to accuracy loss. We introduce IMPACT, an importance-aware activation reconstruction framework that links compression to its effect on model performance. IMPACT formulates compression as an optimization problem that integrates activation structure with gradient-based importance, deriving a closed-form solution where reconstruction bases arise from an importance-weighted activation covariance matrix. This yields low-rank compression explicitly optimized for accuracy preservation. Experiments across multiple models and tasks demonstrate that IMPACT achieves up to 55.4% greater model size reduction while maintaining accuracy comparable to or better than state-of-the-art baselines.

1 Introduction

Large language models (LLMs) have achieved remarkable success across a wide range of domains. However, their massive size poses a significant barrier to deployment, particularly in resource-constrained environments. Larger models require more memory, incur slower token throughput, and demand greater computational and energy resources during inference (Li et al., 2024, 2025b). As a result, there is growing urgency to develop compression techniques that can reduce model size while preserving performance.

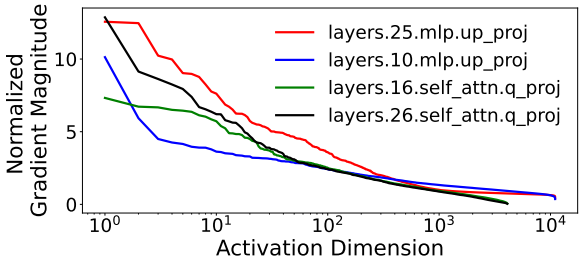


Figure 1: Normalized average gradient magnitudes across activation dimensions in Llama 2-7B on a mathematical reasoning task. For each linear layer, the output activation is a vector $\mathbf{y} \in \mathbb{R}^d$, where each element y_i represents an activation dimension. Dimensions are sorted in descending order of the normalized average gradient magnitudes. Gradient magnitudes vary substantially across activation dimensions—a pattern consistently observed across other models and tasks.

Low-rank weight matrix compression has emerged as a widely used strategy for model compression (Xue et al., 2013; Acharya et al., 2019; Noach and Goldberg, 2020; Huang et al., 2021; Lv et al., 2023; Sharma et al., 2024; Li et al., 2025a). It approximates a weight matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$ as the product of two smaller matrices, $\mathbf{W}_1 \in \mathbb{R}^{m\times k}$ and $\mathbf{W}_2 \in \mathbb{R}^{k\times n}$, where $k \ll m, n$, thereby reducing the number of parameters. Classical methods select $\mathbf{W}_1$ and $\mathbf{W}_2$ to minimize the reconstruction error $\|\mathbf{W} - \mathbf{W}_1\mathbf{W}_2\|$, implicitly assuming that the weight matrix itself is low-rank. However, recent evidence from Yu and Wu (2023) reveals that the weight matrices in large-scale models are often not low-rank, limiting the efficacy of direct weight approximation. Interestingly, they observe that the *activations*—the outputs of linear layers—tend to exhibit much stronger low-rank structure. This has led to a shift in focus: instead of minimizing weight reconstruction error, some recent methods (Yu and

*Equal contribution.
†Corresponding author. Address: 2434 Osborn Dr, Ames, IA 50011, United States. Email: jerryyangli@gmail.com.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 3975–3992

Wu, 2023; Chen et al., 2021b) minimize the error in reconstructing the activations induced by those weights, achieving better empirical performance.

However, minimizing activation reconstruction error alone is insufficient to guarantee strong model performance. As shown in Figure 1, different activation dimensions vary widely in their influence on the loss: dimensions associated with large gradients are highly sensitive to reconstruction errors, whereas others contribute little even when poorly reconstructed. Thus, treating all dimensions equally during compression can disproportionately harm those that matter most—leading to greater performance degradation despite low reconstruction error. This raises a critical question:

How can we align activation reconstruction decisions with their actual performance impact?

Answering this question requires a principled framework that explicitly connects weight compression, activation reconstruction, and their contribution to model performance—a connection that has not yet been systematically explored in the context of low-rank weight matrix compression.

To this end, we propose IMPACT, a theoretical framework that guides importance-aware activation space reconstruction. IMPACT rigorously analyzes the relationship among weight compression, activation reconstruction, and model performance. By explicitly linking reconstruction decisions to their performance impact, it provides principled guidance for selecting weights to minimize performance degradation. The framework is grounded in a formal optimization formulation, which is transformed into a more tractable domain and solved through a rigorous analytical derivation using the Lagrange multiplier method. Despite the complexity of the derivation, the final result is remarkably simple: the optimal activation reconstruction bases are the eigenvectors of an importance-weighted activation covariance matrix $\mathbf{C} = \mathrm{Cov}(\mathbf{y}) \odot \mathbf{M}$, where $\mathrm{Cov}(\mathbf{y})$ is the covariance matrix of the activations, and $\mathbf{M}$ is the gradient-informed importance matrix (Equations (10)–(12)). These eigenvectors yield the weight matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ that minimize performance loss. We apply IMPACT to compress a variety of models across diverse datasets and show that it achieves up to 55.4% greater size reduction while maintaining performance comparable to state-of-the-art baselines.

This paper makes the following contributions:

- We introduce IMPACT, a principled theoretical framework that formally characterizes the relationship between activation reconstruction and its effect on model performance. To our knowledge, this is the first framework within the scope of low-rank weight matrix compression that directly links activation reconstruction choices to model performance.
- We derive a closed-form solution for selecting optimal reconstruction bases using an importance-weighted activation covariance matrix, enabling importance-aware low-rank compression that prioritizes activation dimensions critical to loss minimization.
- We empirically validate IMPACT across a broad range of models and tasks, demonstrating that it achieves substantially greater compression while maintaining performance comparable to state-of-the-art methods.

2 Related Work

Singular value decomposition (SVD) (Golub and Van Loan, 1983) is a widely-used technique for neural network compression, offering low-rank approximations that reduce model size and improve efficiency. Prior work has applied SVD to various network components—convolutional layers, recurrent units, and embeddings—across domains such as language, speech, and vision (Xue et al., 2013; Jaderberg et al., 2014; Denton et al., 2014; Tai et al., 2016; Kim et al., 2016; Lu et al., 2016; Wen et al., 2017; Chen et al., 2018; Acharya et al., 2019; Wang et al., 2021). More recent efforts extend these techniques to Transformer-based models (Noach and Goldberg, 2020; Huang et al., 2021; Li et al., 2023; Lv et al., 2023; Sharma et al., 2024; Li et al., 2025a), compressing attention and feedforward layers to enhance memory and compute efficiency.

Traditional SVD-based methods minimize weight reconstruction error by retaining top singular values and vectors, but this can discard performance-critical information. To address this, FWSVD (Hsu et al., 2022) introduces a weighted factorization scheme guided by Fisher information, assigning higher importance to influential weights.

Recent work has proposed weight compression methods that depart from minimizing weight reconstruction error and instead aim to minimize the reconstruction error of layer activations (Yu and Wu, 2023; Yuan et al., 2023; Chen et al., 2021b). Among them, AFM (Yu and Wu, 2023) explicitly leverages the empirical observation that acti-

vations often exhibit stronger low-rank structure than weights, and optimizes the factorized weights to preserve activations. While these approaches have shown improved empirical results, they typically treat all activation dimensions equally, without fully accounting for their varying contribution to model performance.

In contrast, our work focuses on minimizing the impact of activation reconstruction on model performance. Rather than uniformly reducing reconstruction error, we prioritize preserving the most prediction-critical components of the activation. To our knowledge, this is the first framework in the context of low-rank weight matrix compression that explicitly links activation reconstruction choices to their effect on model accuracy.

Beyond efficiency, another important question is how compression affects trustworthiness. Asante et al. (2026) studies the impact of low-rank compression on privacy, adversarial robustness, ethics, and fairness. More broadly, low-rank compression is only one way to improve inference efficiency; other techniques include, for example, quantization, pruning, and token compression. Sheared LLaMA (Xia et al., 2024) and KiToke (Huang and Li, 2026) illustrate the latter two directions.

3 The IMPACT Framework

In this section, we present IMPACT, our activation reconstruction-based model compression framework. IMPACT identifies a set of directions that enable importance-aware activation reconstruction, minimizing compression-induced performance degradation. We first formulate the optimization problem and define the compression directions in Section 3.1, laying the foundation for performance-preserving reconstruction. Sections 3.2 to 3.6 develop a systematic solution by transforming the activation space, deriving optimal directions via constrained optimization, and constructing the compressed model. Section 3.7 provides a complete algorithm and implementation details of the IMPACT framework.

3.1 Defining the Objective Function

Let $\mathbf{y} \in \mathbb{R}^d$ be the activation produced by a specific layer of the model for a single input sample. We aim to identify a set of r orthonormal vectors $\{\mathbf{u}_1, \ldots, \mathbf{u}_r\}$ that define the directions used to reconstruct activations. Each vector satisfies $\|\mathbf{u}_k\| = 1$ and $\mathbf{u}_i \perp \mathbf{u}_j$ for $i \neq j$, with indices $i, j, k \in \{1, \ldots, r\}$. The reconstructed activation is denoted by $\hat{\mathbf{y}}$.

Our objective is to select $\{\mathbf{u}_k\}$ such that the reconstructed activation $\hat{\mathbf{y}}$ closely approximates the original activation $\mathbf{y}$ while preserving model performance. To this end, we define the following objective function:

$$\min f(\{\mathbf{u}_k\}) = \alpha \mathbb{E}\left[\|\mathbf{y} - \hat{\mathbf{y}}\|^2\right] + \beta \mathbb{E}\left[(\ell(\mathbf{y}) - \ell(\hat{\mathbf{y}}))^2\right] \tag{1}$$

This objective comprises two terms:

- $\alpha \mathbb{E}[\|\mathbf{y} - \hat{\mathbf{y}}\|^2]$ encourages $\hat{\mathbf{y}}$ to be numerically close to $\mathbf{y}$.
- $\beta \mathbb{E}\left[(\ell(\mathbf{y}) - \ell(\hat{\mathbf{y}}))^2\right]$ penalizes changes in the loss function ℓ due to discrepancies between $\mathbf{y}$ and $\hat{\mathbf{y}}$. The values of α and β are set in Section 3.2.

3.2 Bounding the Objective

We now derive an upper bound on the original objective function, providing a more tractable alternative for optimization.

Theorem 1 (Bounding Theorem)**.** *Suppose the loss function ℓ is C^1-smooth, and the activation dimension is d. Then the objective function in Equation (1) is upper bounded by:*

$$f(\{\mathbf{u}_k\}) \leq \mathbb{E}\left[\left\| \sqrt{\beta d\, \mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]^\top} + \alpha \odot (\mathbf{y} - \hat{\mathbf{y}}) \right\|^2\right] \tag{2}$$

Here, $\odot$ denotes the Hadamard (elementwise) product. The proof is presented in Appendix A.2. To simplify the expression and balance the two components of the objective, we set the parameters:

$$\alpha = \eta, \quad \beta = \frac{1 - \eta}{\mathbb{E}\left[\left\|\frac{\partial \ell}{\partial \mathbf{y}}\right\|^2\right]}$$

We study the effect of η in Appendix B.2. Substituting these values into the upper bound yields:

$$h(\{\mathbf{u}_k\}) = \mathbb{E}\left[\left\| \sqrt{\frac{(1-\eta)}{\frac{1}{d}\mathbb{E}\left[\left\|\frac{\partial \ell}{\partial \mathbf{y}}\right\|^2\right]} \mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]^\top} + \eta \odot (\mathbf{y} - \hat{\mathbf{y}}) \right\|^2\right] \tag{3}$$

This gives the inequality:

$$f(\{\mathbf{u}_k\}) \leq h(\{\mathbf{u}_k\})$$

Directly minimizing $f(\{\mathbf{u}_k\})$ under the orthonormal constraints

$$\|\mathbf{u}_k\| = 1, \quad \mathbf{u}_i \perp \mathbf{u}_j \text{ for } i \neq j, \quad i, j, k = 1, \ldots, r$$

is analytically challenging. Solving this problem via mathematical programming is computationally prohibitive due to the high dimensionality of the variables $\{\mathbf{u}_k\}$, which scale with the model's parameter size. To address this challenge, we instead minimize the upper bound $h(\{\mathbf{u}_k\})$ of the original objective, subject to the same orthonormal constraints. This relaxation yields an optimization problem that admits an efficient analytical solution.

3.3 Activation Space Transformation

To optimize the objective $h(\{\mathbf{u}_k\})$ under the orthonormal constraints, we transform the activations into a new space where the optimization problem can be solved analytically. Specifically, we define a transformation coefficient $\mathbf{a}$ as:

$$\mathbf{a} = \sqrt{(1-\eta)\frac{\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]^\top}{\frac{1}{d}\mathbb{E}\left[\left\|\frac{\partial \ell}{\partial \mathbf{y}}\right\|^2\right]} + \eta} \tag{4}$$

Without loss of generality, we assume that the mean-removed reconstructed activations are given by projecting the original activations (after transformation) onto the subspace spanned by $\{\mathbf{u}_k\}$. Formally, we impose:

$$\mathbf{a} \odot (\hat{\mathbf{y}} - \mathbb{E}[\mathbf{y}]) = \left(\sum_k \mathbf{u}_k \mathbf{u}_k^\top\right)(\mathbf{a} \odot (\mathbf{y} - \mathbb{E}[\mathbf{y}])) \tag{5}$$

Defining the transformed activation $\tilde{\mathbf{y}}$ as

$$\tilde{\mathbf{y}} = \mathbf{a} \odot (\mathbf{y} - \mathbb{E}[\mathbf{y}])$$

we can rewrite the objective $h(\{\mathbf{u}_k\})$ in a form that depends only on the transformed activation, as stated in the following theorem.

Theorem 2 (Activation Space Transformation Theorem)**.** *Given the transformation* $\tilde{\mathbf{y}} = \mathbf{a} \odot (\mathbf{y} - \mathbb{E}[\mathbf{y}])$*, the objective* $h(\{\mathbf{u}_k\})$ *becomes:*

$$h(\{\mathbf{u}_k\}) = \mathbb{E}\left[\tilde{\mathbf{y}}^\top \left(\mathbf{I} - \sum_k \mathbf{u}_k \mathbf{u}_k^\top\right)\tilde{\mathbf{y}}\right]$$

The proof is provided in Appendix A.3.

3.4 Lagrange Formulation and Derivation

To solve the constrained optimization problem, we apply the method of Lagrange multipliers. Our goal is to minimize the objective $h(\{\mathbf{u}_k\})$ subject to the normalization constraint $\|\mathbf{u}_k\| = 1$, along with the orthogonality constraints. To enforce this, we define the Lagrangian function:

$$L(\{\mathbf{u}_k\}) = h(\{\mathbf{u}_k\}) + \sum_{k=1}^{r} \lambda_k \left(\mathbf{u}_k^\top \mathbf{u}_k - 1\right) \tag{6}$$

where λ_k is the Lagrange multiplier associated with the constraint $\mathbf{u}_k^\top \mathbf{u}_k = 1$.

We derive the optimality conditions by taking the derivative of $L(\{\mathbf{u}_k\})$ with respect to each $\mathbf{u}_k$ and setting it to zero. Using standard results from matrix calculus, we obtain:

$$\frac{\partial L}{\partial \mathbf{u}_k} = -2\mathbf{u}_k^\top \mathbb{E}\left[\tilde{\mathbf{y}}\tilde{\mathbf{y}}^\top\right] + \lambda_k \mathbf{u}_k^\top \tag{7}$$

To simplify notations, we define the *importance-weighted activation covariance matrix*:

$$\mathbf{C} = \mathbb{E}\left[\tilde{\mathbf{y}}\tilde{\mathbf{y}}^\top\right] \tag{8}$$

Substituting into Equation (7), the optimality condition becomes:

$$-2\mathbf{u}_k^\top \mathbf{C} + \lambda_k \mathbf{u}_k^\top = 0 \tag{9}$$

The following theorem characterizes the structure of $\mathbf{C}$.

Theorem 3 (Importance-Weighted Activation Covariance Matrix)**.** *The matrix* $\mathbf{C}$ *is equal to the Hadamard product of the activation covariance matrix* $\mathrm{Cov}(\mathbf{y})$ *and the gradient-informed importance matrix* $\mathbf{M}$*, i.e.,*

$$\mathbf{C} = \mathrm{Cov}(\mathbf{y}) \odot \mathbf{M} \tag{10}$$

where

$$\mathrm{Cov}(\mathbf{y}) = \mathbb{E}\left[(\mathbf{y} - \mathbb{E}[\mathbf{y}])(\mathbf{y} - \mathbb{E}[\mathbf{y}])^\top\right] \tag{11}$$

$$\mathbf{M} = \mathbf{a}\mathbf{a}^\top \tag{12}$$

A detailed derivation is provided in Appendix A.4.

Remark 1. Since both $\mathrm{Cov}(\mathbf{y})$ and $\mathbf{M}$ are positive semidefinite, their Hadamard product $\mathbf{C}$ is also positive semidefinite by the Schur product theorem (Zhang, 2005).

Remark 2. Because $\mathrm{Cov}(\mathbf{y})$ and $\mathbf{M}$ are symmetric, $\mathbf{C}$ is symmetric as well.

To illustrate the importance-aware activation reconstruction enabled by the gradient-informed importance matrix $\mathbf{M}$, we present visual heatmaps and detailed analysis in Appendix B.1.

3.5 Reconstruction Direction

From Equation (9) and the fact that $\mathbf{C}$ is real and symmetric, we have:

$$\mathbf{C}\mathbf{u}_k = \lambda_k \mathbf{u}_k \tag{13}$$

This implies that each reconstruction direction $\mathbf{u}_k$ is an eigenvector of $\mathbf{C}$. We formalize this result in the following theorem.

Theorem 4 (Reconstruction Direction Theorem)**.** *To minimize the objective $h(\{\mathbf{u}_k\})$ under orthonormality constraints, the optimal k^{th} reconstruction direction $\mathbf{u}_k$ is the eigenvector corresponding to the k^{th} largest eigenvalue of the importance-weighted activation covariance matrix $\mathbf{C}$.*

A full derivation is provided in Appendix A.5.

The reconstruction directions $\{\mathbf{u}_k\}_{k=1}^r$ are obtained by selecting and normalizing the top r eigenvectors of $\mathbf{C}$. These eigenvectors are guaranteed to be orthogonal.

3.6 Compressed Model Representation

After obtaining the reconstruction directions $\{\mathbf{u}_k\}_{k=1}^r$, we construct the compressed model accordingly.

Given the relationship between the original activation $\mathbf{y}$ and the reconstructed activation $\hat{\mathbf{y}}$:

$$\mathbf{a} \odot (\hat{\mathbf{y}} - \mathbb{E}[\mathbf{y}]) = \left(\sum_k \mathbf{u}_k \mathbf{u}_k^\top \right) (\mathbf{a} \odot (\mathbf{y} - \mathbb{E}[\mathbf{y}]))$$

and the fact that $\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b}$, where $\mathbf{W}$ and $\mathbf{b}$ are the original layer's weight matrix and bias, we can express the reconstructed activation $\hat{\mathbf{y}}$ as follows:

Theorem 5 (Activation Reconstruction Theorem)**.** *The reconstructed activation $\hat{\mathbf{y}}$, which satisfies the projection condition*

$$\mathbf{a} \odot (\hat{\mathbf{y}} - \mathbb{E}[\mathbf{y}]) = \left(\sum_k \mathbf{u}_k \mathbf{u}_k^\top \right) (\mathbf{a} \odot (\mathbf{y} - \mathbb{E}[\mathbf{y}]))$$

is given by:

$$\hat{\mathbf{y}} = \left[\mathbf{U} \oslash (\mathbf{a} \cdot \mathbf{1}_r^\top) \right] \left[(\mathbf{U} \odot (\mathbf{a} \cdot \mathbf{1}_r^\top))^\top \mathbf{W} \right] \mathbf{x} + \mathbb{E}[\mathbf{y}] + (\mathbf{U}\mathbf{U}^\top \odot (\frac{1}{\mathbf{a}} \cdot \mathbf{a}^\top))(\mathbf{b} - \mathbb{E}[\mathbf{y}])$$

where $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_r]$, $\mathbf{1}_r$ is an r-dimensional column vector of ones, $\mathbf{W}$ and $\mathbf{b}$ are the original layer's weight matrix and bias, $\mathbf{x}$ is the input activation, and $\oslash$ denotes element-wise division.

A full derivation is provided in Appendix A.6.

Based on this result, the compressed layer is implemented using two linear layers:

- The first layer has a weight matrix

$$\mathbf{W}_1 = (\mathbf{U} \odot (\mathbf{a} \cdot \mathbf{1}_r^\top))^\top \mathbf{W}$$

  and no bias;

- The second layer has a weight matrix

$$\mathbf{W}_2 = \mathbf{U} \oslash (\mathbf{a} \cdot \mathbf{1}_r^\top)$$

  and bias

$$\mathbf{b}' = \mathbb{E}[\mathbf{y}] + (\mathbf{U}\mathbf{U}^\top \odot (\frac{1}{\mathbf{a}} \cdot \mathbf{a}^\top))(\mathbf{b} - \mathbb{E}[\mathbf{y}])$$

The compressed layer is expressed as:

$$\hat{\mathbf{y}} = \mathbf{W}_2(\mathbf{W}_1\mathbf{x}) + \mathbf{b}'$$

3.7 IMPACT Algorithm Description

Algorithm 1 outlines the procedure, which consists of two stages: profiling and compression.

3.7.1 Profiling Stage

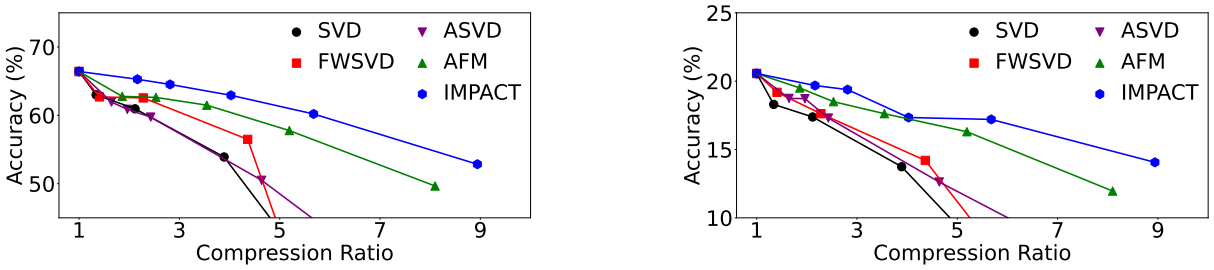
The algorithm gathers activation and gradient statistics for each linear layer in the model. Specifically, it computes the mean activation, the activation co-

Algorithm 1: IMPACT Algorithm

Input: Model $\mathcal{LM}$
Output: Compressed Model $\mathcal{LM}'$
Data: Dataset D, Keeping Ratio k

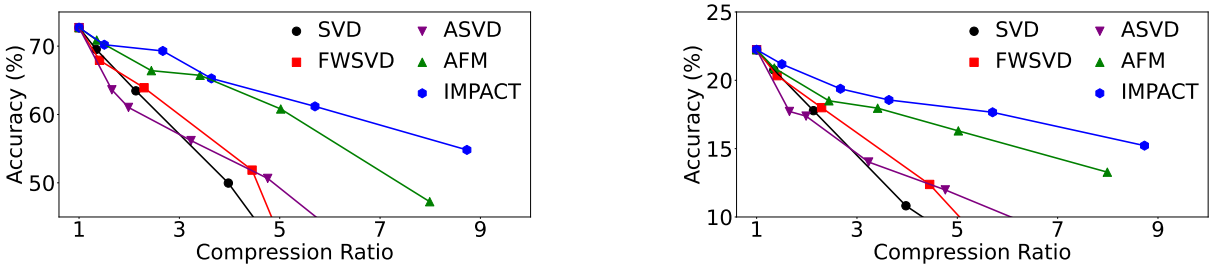
Stage 1: Profiling;
Let n be the total number of samples in D;
for *each layer l in $\mathcal{LM}$* **do**
  Initialize
  $\mathbb{E}\left[\mathbf{y}\mathbf{y}^\top\right]_l = 0$, $\mathbb{E}[\mathbf{y}]_l = 0$, $\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]_l = 0$;
for *each sample $s \in D$* **do**
  Get activation $\mathbf{y}_l$ and gradient $\frac{\partial \ell}{\partial \mathbf{y}_l}$ for layer l;
  for *each layer l in $\mathcal{LM}$* **do**
    $\mathbb{E}\left[\mathbf{y}\mathbf{y}^\top\right]_l \leftarrow \mathbb{E}\left[\mathbf{y}\mathbf{y}^\top\right]_l + \mathbf{y}_l\mathbf{y}_l^\top$;
    $\mathbb{E}[\mathbf{y}]_l \leftarrow \mathbb{E}[\mathbf{y}]_l + \mathbf{y}_l$;
    $\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]_l \leftarrow \mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]_l + \left(\frac{\partial \ell}{\partial \mathbf{y}_l}\right)^2$;
for *each layer l in $\mathcal{LM}$* **do**
  $\mathbb{E}\left[\mathbf{y}\mathbf{y}^\top\right]_l \leftarrow \mathbb{E}\left[\mathbf{y}\mathbf{y}^\top\right]_l / n$;
  $\mathbb{E}[\mathbf{y}]_l \leftarrow \mathbb{E}[\mathbf{y}]_l / n$;
  $\mathrm{Cov}(\mathbf{y})_l \leftarrow \mathbb{E}\left[\mathbf{y}\mathbf{y}^\top\right]_l - \mathbb{E}[\mathbf{y}]_l\,\mathbb{E}[\mathbf{y}]_l^\top$;

Stage 2: Compression;
for *each layer l in $\mathcal{LM}$* **do**
  // For brevity, the subscript l is omitted from the notations presented below.
  Compute the transformation coefficient $\mathbf{a}$ based on Equation (4);
  Compute the gradient-informed importance matrix $\mathbf{M}$ based on Equation (12);
  Compute the importance-weighted activation covariance matrix $\mathbf{C}$ based on Equation (10);
  $[\mathbf{U}, \mathbf{\Lambda}] = \text{eigenvalue\_decomposition}(\mathbf{C})$;
  // The columns of $\mathbf{U}$ are the eigenvectors of $\mathbf{C}$;
  // The vector $\mathbf{\Lambda}$ consists of the eigenvalues of $\mathbf{C}$;
  Sort the elements of $\mathbf{\Lambda}$ in descending order and reorder $\mathbf{U}$ accordingly;
  Find smallest r such that $\left(\sum_{j=1}^r \sqrt{\Lambda_j}\right) / \left(\sum_{j=1}^d \sqrt{\Lambda_j}\right) \geq k/100$;
  $\mathbf{U} \leftarrow$ First r columns of $\mathbf{U}$;
  Substitute the original linear layer with two new linear layers with smaller sizes:
  The first new layer has a weight matrix of $(\mathbf{U} \odot (\mathbf{a} \cdot \mathbf{1}_r^\top))^\top \mathbf{W}$ and no bias;
  The second new layer has a weight matrix of $\mathbf{U} \oslash (\mathbf{a} \cdot \mathbf{1}_r^\top)$ and a bias of $\mathbb{E}[\mathbf{y}] + (\mathbf{U}\mathbf{U}^\top \odot (\frac{1}{\mathbf{a}} \cdot \mathbf{a}^\top))(\mathbf{b} - \mathbb{E}[\mathbf{y}])$;
return Compressed Model $\mathcal{LM}'$;



(a) GSM8K (b) MATH

Figure 2: Pass@1 accuracy and model size of Llama 2-7B compressed with various low-rank algorithms on the mathematical reasoning task. Exact values are listed in Table 6.



(a) GSM8K (b) MATH

Figure 3: Pass@1 accuracy and model size of Llama 2-13B compressed with various low-rank algorithms on the mathematical reasoning task. Exact values are listed in Table 7.

variance matrix, and the mean squared gradient with respect to the activations. These statistics form the basis for the subsequent compression step.

3.7.2 Compression Stage

Using the collected statistics, the algorithm constructs the importance-weighted activation covariance matrix $\mathbf{C}$ for each linear layer by applying a Hadamard product between the activation covariance and the gradient-informed importance matrix. Eigenvalue decomposition is then performed on $\mathbf{C}$ to extract the top eigenvectors, which define the compression directions. Each original linear layer is subsequently replaced by a pair of smaller linear layers designed to preserve model performance.

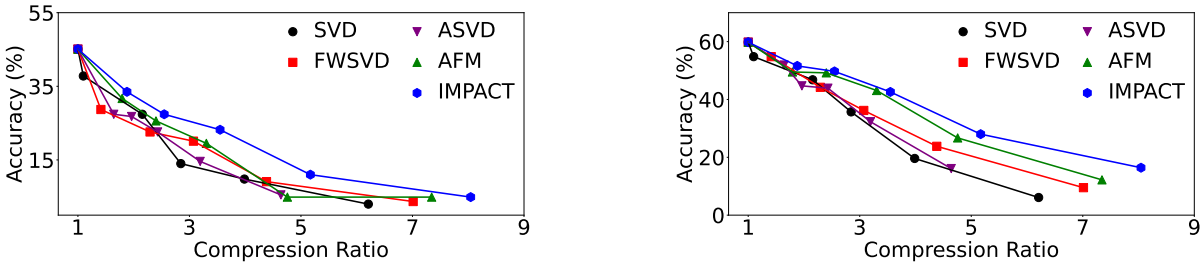
4 Experiments

4.1 Evaluation Methodology

We evaluate the effectiveness of low-rank compression algorithms on two tasks: mathematical reasoning and code generation. For mathematical reasoning, we use the Llama 2-7B and -13B models (Touvron et al., 2023); for code generation, we use CodeLlama-7B and -13B (Rozière et al., 2023). Each model is first finetuned on a task-specific finetuning set, then compressed using a low-rank method, and finally undergoes post-compression finetuning before evaluation. For fair comparison, we applied the same supervised fine-tuning (SFT) protocol to all methods, including IMPACT and all baselines. We used AdamW with a learning rate of 2e-5 for Llama 2-7B and CodeLlama-7B, and 1e-5 for Llama 2-13B and CodeLlama-13B, along with cosine learning rate scheduling (warmup ratio 3%), BF16 mixed precision, and FSDP. The total batch size was 128 for both Llama 2-7B and CodeLlama-7B (2 GPUs × 64 samples per GPU), and for Llama 2-13B and CodeLlama-13B (4 GPUs × 32 samples per GPU). All profiling was performed as a one-time post-training pass on a single NVIDIA A100 GPU. Additional details on the profiling cost and stability with respect to calibration data are provided in Appendix B.3.

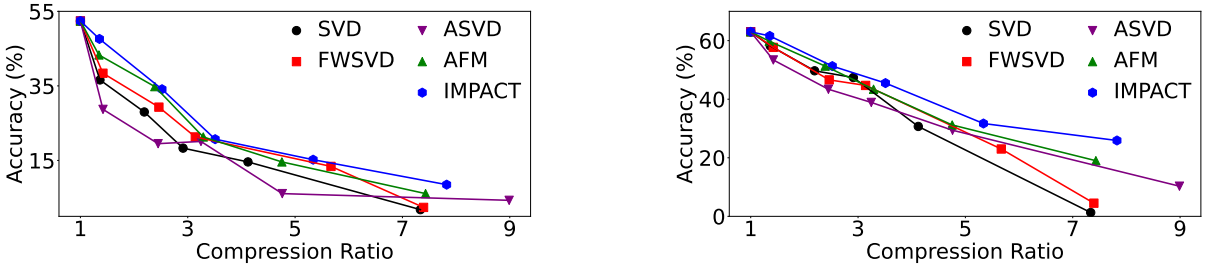
We employ distinct datasets for calibration and evaluation. For calibration, we use MetaMathQA-395K (Yu et al., 2024) for mathematical reasoning and Code-Instructions-120K (Bisht, 2023) for code generation. For evaluation, on the mathematical reasoning task, we evaluate on GSM8K (Cobbe et al., 2021) and Hendrycks' MATH (Hendrycks et al., 2021); on code generation, we use MBPP (Austin et al., 2021) and HumanEval (Chen et al., 2021a) as evaluation sets.

We compare our proposed method, IMPACT,



(a) HumanEval (b) MBPP

Figure 4: Pass@1 accuracy and model size of CodeLlama-7B compressed with various low-rank algorithms on the code generation task. Exact values are listed in Table 8.



(a) HumanEval (b) MBPP

Figure 5: Pass@1 accuracy and model size of CodeLlama-13B compressed with various low-rank algorithms on the code generation task. Exact values are listed in Table 9.

against state-of-the-art low-rank compression techniques, including SVD (Xue et al., 2013; Wang et al., 2021; Noach and Goldberg, 2020; Huang et al., 2021; Li et al., 2023; Lv et al., 2023; Sharma et al., 2024; Lin et al., 2025), a widely used matrix factorization method; FWSVD (Hsu et al., 2022), which incorporates weight importance; and activation-aware approaches such as ASVD (Yuan et al., 2023) and AFM (Yu and Wu, 2023).

Beyond low-rank compression methods, we also benchmark IMPACT against compression techniques from other paradigms, including QLoRA (Dettmers et al., 2023), a quantization approach that finetunes low-rank adapters, and FLAP (An et al., 2024), a pruning method that removes weights based on magnitude and activation variance. These comparisons highlight IMPACT's robustness and effectiveness across diverse compression strategies.

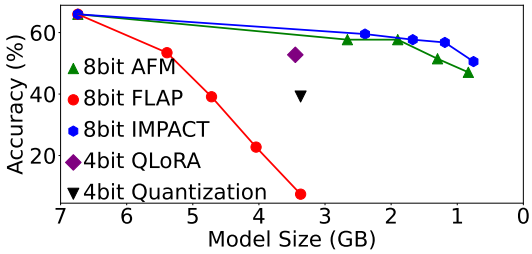
4.2 Evaluation of Low-Rank Compression for Mathematical Reasoning

Figures 2 and 3 show the performance of low-rank compression methods on Llama 2-7B and -13B for the mathematical reasoning task. We evaluate the Pass@1 accuracy of the models across a range of compression ratios (the ratio of the original model size to the compressed model size). The precise numerical values corresponding to these performance curves are reported in Tables 6 and 7. Our proposed method, IMPACT, consistently outperforms SVD, FWSVD, ASVD, and AFM across all compression ratios, achieving greater compression while maintaining comparable or superior accuracy.

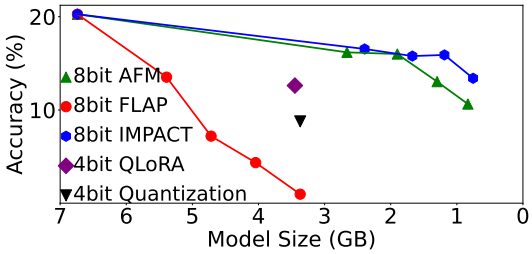
On Llama 2-7B, IMPACT achieves up to 55.4% greater size reduction than the strongest baseline (AFM) on GSM8K, and up to 31.7% more on MATH, while maintaining the same accuracy. Across all evaluated compression ratios, it compresses the model over 40% more than SVD, FWSVD, and ASVD on both datasets while delivering similar or better performance. Similar patterns are observed for Llama 2-13B, where IMPACT achieves up to 39.8% more compression than AFM on GSM8K and 36.5% more on MATH. At compression ratios above 2.5×, IMPACT continues to deliver over 35% more compression than SVD, FWSVD, and ASVD while maintaining better performance.

4.3 Evaluation of Low-Rank Compression for Code Generation

Figures 4 and 5 show the performance of IMPACT and baselines on CodeLlama-7B and -13B for code generation, with precise numerical values reported in Tables 8 and 9. We evaluate the Pass@1 accuracy

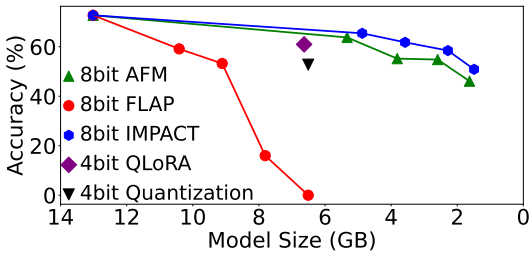


(a) GSM8K

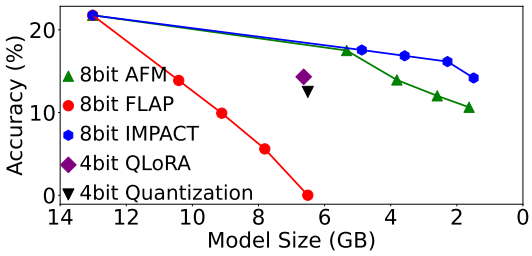


(b) MATH

Figure 6: Pass@1 accuracy and model size of Llama 2-7B models compressed using quantization alone, as well as in combination with low-rank compression or pruning techniques, evaluated on the mathematical reasoning task.



(a) GSM8K



(b) MATH

Figure 7: Pass@1 accuracy and model size of Llama 2-13B models compressed using quantization alone, as well as in combination with low-rank compression or pruning techniques, evaluated on the mathematical reasoning task.

on the HumanEval and MBPP benchmarks across a range of compression ratios.

IMPACT consistently outperforms baseline methods by achieving greater compression while maintaining comparable or superior accuracy on both code generation tasks. On CodeLlama-7B, IMPACT reduces model size by up to 24.2% more than the best-performing baseline on HumanEval, and by 30.7% more on MBPP. Similar trends are observed on CodeLlama-13B, where IMPACT achieves up to 18.1% more compression on HumanEval and 28.0% more on MBPP compared to the strongest baseline.

4.4 Integrating IMPACT with Quantization

Quantization and low-rank compression are distinct model compression techniques grounded in different principles: quantization reduces the precision of model weights, whereas low-rank compression approximates weight matrices as the product of smaller matrices. Quantization generally preserves performance at 8-bit precision or higher but often degrades accuracy at lower precisions like 4-bit. To assess the combined effect of quantization and other compression methods such as low-rank compression and pruning, we integrate 8-bit quantization with IMPACT, AFM (the strongest low-rank baseline), and FLAP to produce compressed models of various sizes.

Results on mathematical reasoning with Llama 2-7B (Figure 6, Tables 1 and 3) show that IMPACT with 8-bit quantization consistently outperforms pure 4-bit quantization, 4-bit QLoRA, 8-bit AFM, and 8-bit FLAP. For example, 8-bit IMPACT achieves higher accuracy at a model size of 1.19 GB than 8-bit FLAP, 8-bit AFM, and 4-bit QLoRA do at 5.39 GB, 1.30 GB, and 3.45 GB, respectively. Similar trends are observed for Llama 2-13B (Figure 7, Tables 2 and 3), where 8-bit IMPACT outperforms all baselines. These results highlight the superior performance of IMPACT and the benefit of combining low-rank compression with quantization, which yields higher accuracy than either technique alone at the same model size.

4.5 Inference Performance

To evaluate inference performance, we measure the throughput and memory usage of compressed models on the mathematical reasoning task. Figure 8 presents results for models compressed using SVD, FWSVD, AFM, ASVD, and IMPACT across various model sizes. As expected, larger models exhibit lower throughput and higher memory consumption across all methods. When model size is held constant, all approaches demonstrate comparable throughput and memory consumption. How-

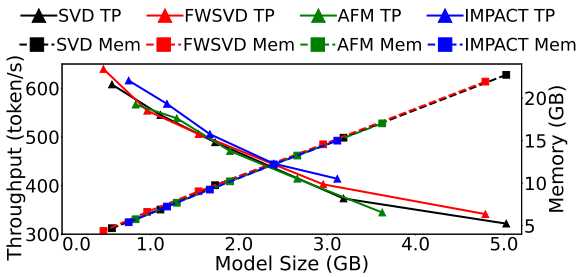


Figure 8: Throughput and memory consumption of compressed models. Exact values are provided in Table 14.

ever, because IMPACT achieves similar accuracy at smaller model sizes, it delivers higher throughput and lower memory usage at equivalent accuracy levels. In particular, compared to AFM—the strongest baseline—IMPACT improves throughput by up to 41% and reduces memory use by up to 42%.

5 Conclusion

This paper introduces IMPACT, a principled framework for low-rank model compression that explicitly links activation reconstruction to model performance. In contrast to prior methods that either compress weights directly or minimize activation reconstruction error, IMPACT guides activation reconstruction along directions most critical to model behavior. By formulating and solving a well-grounded optimization problem, we derive a closed-form solution in which the optimal reconstruction bases are the eigenvectors of an importance-weighted activation covariance matrix.

Our empirical results across multiple LLMs and multiple benchmarks demonstrate that IMPACT consistently achieves greater compression—up to 55.4% more than the state-of-the-art—while maintaining similar accuracy. These findings not only validate the theoretical underpinnings of our method but also highlight its practical effectiveness for real-world deployment.

8-bit FLAP	Model Size (GB)	6.74	5.39	4.72	4.04	3.37
	GSM8K Acc (%)	66.0	53.4	39.1	22.7	7.4
	MATH Acc (%)	20.3	13.5	7.2	4.4	1.0
8-bit AFM	Model Size (GB)	6.74	2.66	1.90	1.30	0.83
	GSM8K Acc (%)	66.0	57.7	57.7	51.5	47.1
	MATH Acc (%)	20.3	16.2	16.0	13.0	10.6
8-bit IMPACT	Model Size (GB)	6.74	2.40	1.67	1.19	0.75
	GSM8K Acc (%)	66.0	59.5	57.7	56.8	50.6
	MATH Acc (%)	20.3	16.5	15.8	15.9	13.4

Table 1: Pass@1 accuracy and model size of 8-bit-quantized Llama 2-7B models compressed by FLAP, AFM, and IMPACT for mathematical reasoning. The results for 8-bit FLAP are taken from Li et al. (2025a).

8-bit FLAP	Model Size (GB)	13.02	10.41	9.11	7.81	6.51
	GSM8K Acc (%)	72.7	59.1	53.2	16.0	0.0
	MATH Acc (%)	21.8	13.9	9.9	5.6	0.0
8-bit AFM	Model Size (GB)	13.02	5.33	3.81	2.59	1.63
	GSM8K Acc (%)	72.7	63.8	55.2	54.8	46.1
	MATH Acc (%)	21.8	17.5	13.9	12.0	10.6
8-bit IMPACT	Model Size (GB)	13.02	4.87	3.58	2.28	1.49
	GSM8K Acc (%)	72.7	65.4	61.8	58.5	50.9
	MATH Acc (%)	21.8	17.6	16.9	16.2	14.2

Table 2: Pass@1 accuracy and model size of 8-bit-quantized Llama 2-13B models compressed by FLAP, AFM, and IMPACT for mathematical reasoning.

Model Variant	Model Size (GB)	Task	Accuracy (%)
4-bit-quantized 7B	3.37	GSM8K	39.2
		MATH	8.8
4-bit-QLoRA 7B	3.45	GSM8K	54.1
		MATH	12.6
4-bit-quantized 13B	6.51	GSM8K	52.8
		MATH	12.5
4-bit-QLoRA 13B	6.63	GSM8K	61.0
		MATH	14.3

Table 3: Pass@1 accuracy and model size of Llama 2-7B and -13B quantized using standard 4-bit quantization and 4-bit QLoRA on the mathematical reasoning task. Part of the results is taken from Li et al. (2025a).

IMPACT offers a general and extensible foundation for future compression research. By establishing a formal link between compression decisions and performance outcomes, our work provides both insight and actionable tools for efficient LLM deployment—advancing the broader goal of making powerful models more accessible and sustainable.

Limitations

This work establishes a connection between activation reconstruction and model performance, enabling performance-aware activation space reconstruction for model compression. Model compression is a crucial topic in natural language processing, influencing the deployment and applicability of NLP models. We believe that our approach, which links model design decisions to performance, has the potential to extend beyond model compression and be applied to other areas. We leave exploration of this broader applicability to future work.

Acknowledgments

This work is supported by a faculty startup grant from Iowa State University. Computational resources are provided by the HPC of the university, including equipment funded by NSF under MRI Grant Nos. 1726447 and 2018594.

References

Anish Acharya, Rahul Goel, Angeliki Metallinou, and Inderjit Dhillon. 2019. Online Embedding Compression for Text Classification Using Low Rank Matrix Factorization. In *AAAI Conference on Artificial Intelligence*.

Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. 2024. Fluctuation-based Adaptive Structured Pruning for Large Language Models. In *AAAI Conference on Artificial Intelligence*.

Daniel Agyei Asante, Md Mokarram Chowdhury, and Yang Li. 2026. Decomposed Trust: Privacy, Adversarial Robustness, Ethics, and Fairness in Low-Rank LLMs. In *Findings of the Association for Computational Linguistics: ACL*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program Synthesis with Large Language Models. *arXiv preprint arXiv:2108.07732*.

Tarun Bisht. 2023. Code Instructions 120K Alpaca. https://huggingface.co/datasets/iamtarun/code_instructions_120k_alpaca. Accessed: 2026-01-01.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021a. Evaluating Large Language Models Trained on Code. *arXiv preprint arXiv:2107.03374*.

Patrick Chen, Si Si, Yang Li, Ciprian Chelba, and Cho-Jui Hsieh. 2018. GroupReduce: Block-wise Low-Rank Approximation for Neural Language Model Shrinking. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Patrick Chen, Hsiang-Fu Yu, Inderjit Dhillon, and Cho-Jui Hsieh. 2021b. DRONE: Data-aware Low-Rank Compression for Large NLP Models. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*.

Emily Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. 2014. Exploiting Linear Structure within Convolutional Networks for Efficient Evaluation. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Gene H. Golub and Charles F. Van Loan. 1983. *Matrix Computations*. Johns Hopkins University Press.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring Mathematical Problem Solving with the MATH Dataset. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. 2022. Language Model Compression with Weighted Low-Rank Factorization. In *International Conference on Learning Representation (ICLR)*.

Haifeng Huang and Yang Li. 2026. KiToke: Kernel-based Interval-aware Token Compression for Video Large Language Models. *arXiv preprint arXiv:2604.03414*.

Shaoyi Huang, Shiyang Chen, Hongwu Peng, Daniel Manu, Zhenglun Kong, Geng Yuan, Lei Yang, Shusen Wang, Hang Liu, and Caiwen Ding. 2021. HMC-TRAN: A Tensor-core Inspired Hierarchical Model Compression for Transformer-based DNNs on GPU. In *Great Lakes Symposium on VLSI (GLSVLSI)*.

Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. 2014. Speeding up Convolutional Neural Networks with Low Rank Expansions. In *British Machine Vision Conference (BMVC)*.

Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang, and Dongjun Shin. 2016. Compression of Deep Convolutional Neural Networks for Fast and Low Power Mobile Applications. In *International Conference on Learning Representations (ICLR)*.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. 2019. SNIP: Single-Shot Network Pruning Based on Connection Sensitivity. In *International Conference on Learning Representations (ICLR)*.

Hailong Li, Jaewan Choi, Yongsuk Kwon, and Jung Ho Ahn. 2023. A Hardware-Friendly Tiled Singular-Value Decomposition-Based Matrix Multiplication for Transformer-Based Models. *IEEE Computer Architecture Letters (CAL)*, 22:169–172.

Yang Li, Daniel Agyei Asante, Changsheng Zhao, Ernie Chang, Yangyang Shi, and Vikas Chandra. 2025a. Streamlining Language Models via Semantic Basis Analysis. *Transactions on Machine Learning Research (TMLR)*.

Yang Li, Liangzhen Lai, Yuan Shangguan, Forrest N. Iandola, Ernie Chang, Yangyang Shi, and Vikas Chandra. 2024. Folding Attention: Memory and

Power Optimization for On-Device Transformer-based Streaming Speech Recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Yang Li, Yuan Shangguan, Yuhao Wang, Liangzhen Lai, Ernie Chang, Changsheng Zhao, Yangyang Shi, and Vikas Chandra. 2025b. Breaking Down Power Barriers in On-Device Streaming ASR: Insights and Solutions. In *Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics (NAACL), Industry Track*.

Chi-Heng Lin, Shangqian Gao, James Seale Smith, Abhishek Patel, Shikhar Tuli, Yilin Shen, Hongxia Jin, and Yen-Chang Hsu. 2025. MoDeGPT: Modular Decomposition for Large Language Model Compression. In *International Conference on Learning Representations (ICLR)*.

Zhiyun Lu, Vikas Sindhwani, and Tara N Sainath. 2016. Learning Compact Recurrent Neural Networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Xiuqing Lv, Peng Zhang, Sunzhu Li, Guobing Gan, and Yueheng Sun. 2023. LightFormer: Light-weight Transformer Using SVD-based Weight Transfer and Parameter Sharing. In *Findings of the Association for Computational Linguistics: ACL 2023*.

Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. 2019. Importance Estimation for Neural Network Pruning. In *IEEE/CVF conference on computer vision and pattern recognition (CVPR)*.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2017. Pruning Convolutional Neural Networks for Resource Efficient Inference. In *International Conference on Learning Representations (ICLR)*.

Matan Ben Noach and Yoav Goldberg. 2020. Compressing Pre-trained Language Models by Matrix Decomposition. In *1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing (AACL-IJCNLP)*.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, and 7 others. 2023. Code Llama: Open Foundation Models for Code. *arXiv preprint arXiv:2308.12950*.

Pratyusha Sharma, Jordan T. Ash, and Dipendra Misra. 2024. The Truth is in there: Improving Reasoning in Language Models with Layer-Selective Rank Reduction. In *International Conference on Learning Representations (ICLR)*.

Cheng Tai, Tong Xiao, Yi Zhang, Xiaogang Wang, and E. Weinan. 2016. Convolutional Neural Networks With Low-rank Regularization. In *International Conference on Learning Representations (ICLR)*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. Llama 2: Open Foundation and Finetuned Chat Models. *arXiv preprint arXiv:2307.09288*.

Hongyi Wang, Saurabh Agarwal, and Dimitris Papailiopoulos. 2021. Pufferfish: Communication-efficient Models at No Extra Cost. In *Conference on Machine Learning and Systems (MLSys)*.

Wei Wen, Cong Xu, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2017. Coordinating Filters for Faster Deep Neural Networks. In *IEEE International Conference on Computer Vision (ICCV)*.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2024. Sheared LLaMA: Accelerating Language Model Pre-training via Structured Pruning. In *International Conference on Learning Representations (ICLR)*.

Jian Xue, Jinyu Li, and Yifan Gong. 2013. Restructuring of Deep Neural Network Acoustic Models with Singular Value Decomposition. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

Hao Yu and Jianxin Wu. 2023. Compressing Transformers: Features Are Low-Rank, But Weights Are Not! In *AAAI Conference on Artificial Intelligence*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. In *International Conference on Learning Representation (ICLR)*.

Zhihang Yuan, Yuzhang Shang, Yue Song, Qiang Wu, Yan Yan, and Guangyu Sun. 2023. ASVD: Activation-aware Singular Value Decomposition for Compressing Large Language Models. *arXiv preprint arXiv:2312.05821*.

Fuzhen Zhang. 2005. *The Schur Complement and Its Applications*. Springer.

A Theoretical Derivation

A.1 Mathematical Preliminaries

We first introduce key mathematical definitions and properties that serve as the foundation for our derivation.

Definition 1 (Differentiation Convention)**.** *For a differentiable function $\ell(\mathbf{y}) : \mathbb{R}^n \to \mathbb{R}$ where $\mathbf{y} = [y_1, \ldots, y_n]^\top \in \mathbb{R}^n$, the derivative of ℓ with respect*

to $\mathbf{y}$ *following the denominator-layout convention is given by the row vector*

$$\frac{\partial \ell}{\partial \mathbf{y}} = \begin{bmatrix} \frac{\partial \ell}{\partial y_1} & \cdots & \frac{\partial \ell}{\partial y_n} \end{bmatrix}$$

We maintain this convention throughout our derivation.

Definition 2 (Hadamard Product)**.** *Given two vectors* $\mathbf{p} = [p_1, \ldots, p_n]^\top \in \mathbb{R}^n$ *and* $\mathbf{q} = [q_1, \ldots, q_n]^\top \in \mathbb{R}^n$*, the Hadamard product (element-wise product) is defined as*

$$\mathbf{p} \odot \mathbf{q} = \begin{bmatrix} p_1 q_1 \\ \vdots \\ p_n q_n \end{bmatrix} \in \mathbb{R}^n$$

Definition 3 (Orthogonality and Normalization)**.** *Given column vectors* $\mathbf{u}_i, \mathbf{u}_j \in \mathbb{R}^n$*, their orthogonality and normalization properties are defined as follows:*

- *Vectors* $\mathbf{u}_i$ *and* $\mathbf{u}_j$ *are orthogonal if their inner product satisfies*

$$\mathbf{u}_i^\top \mathbf{u}_j = 0$$

- *A vector* $\mathbf{u}_i$ *is normalized if*

$$\mathbf{u}_i^\top \mathbf{u}_i = \|\mathbf{u}_i\|^2 = 1$$

Property 1 (QM-AM Inequality)**.** *For the set* $\{y_i \mid y_i \in \mathbb{R}_{\geq 0},\ i = 1, \cdots, n\}$*, the arithmetic mean (AM) and the quadratic mean (QM) are defined as:*

$$\mathrm{AM} = \frac{1}{n}\sum_{i=1}^{n} y_i, \qquad \mathrm{QM} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} y_i^2}$$

The QM-AM inequality states that: $\mathrm{QM} \geq \mathrm{AM}$*, with equality if and only if* $y_1 = \cdots = y_n$*.*

Method 1 (Lagrange Multiplier Method)**.** *The Lagrange multiplier method determines the local extrema of a function under explicit functional constraints. Given an objective function* $f : \mathbb{R}^n \to \mathbb{R}$ *and a constraint function* $g : \mathbb{R}^n \to \mathbb{R}$*, where the constraint is given by* $g(x) = 0$*, the Lagrangian function is defined as:*

$$\mathcal{L}(x, \lambda) = f(x) + \lambda g(x)$$

where $\lambda \in \mathbb{R}$ *is the Lagrange multiplier. The optimal solution is obtained by solving the system of equations:*

$$\frac{d}{dx}\mathcal{L}(x, \lambda) = 0, \quad \frac{d}{d\lambda}\mathcal{L}(x, \lambda) = 0$$

A.2 Bounding Theorem

Theorem 1. *Suppose the loss function* ℓ *is* C^1*-smooth and the activation dimension is* d*. Then the objective function*

$$f(\{\mathbf{u}_k\}) = \alpha \mathbb{E}\big[\|\mathbf{y} - \hat{\mathbf{y}}\|^2\big] + \beta \mathbb{E}\big[(\ell(\mathbf{y}) - \ell(\hat{\mathbf{y}}))^2\big] \quad (14)$$

is upper bounded by:

$$f(\{\mathbf{u}_k\}) \leq \mathbb{E}\left[\left\| \sqrt{\beta d\, \mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]^\top} + \alpha \odot (\mathbf{y} - \hat{\mathbf{y}}) \right\|^2\right]$$

Proof. Performing Taylor expansion[1] on the loss function, we obtain:

$$\ell(\hat{\mathbf{y}}) \approx \ell(\mathbf{y}) + \frac{\partial \ell}{\partial \mathbf{y}}(\hat{\mathbf{y}} - \mathbf{y})$$

The higher-order terms (e.g., second-order terms and beyond) are ignored because they are computationally expensive to estimate and difficult to capture accurately in practical applications. Plugging this into Equation (14), we get the following results:

$$\begin{aligned} f(\{\mathbf{u}_k\}) &\approx \alpha \mathbb{E}\big[\|\mathbf{y} - \hat{\mathbf{y}}\|^2\big] + \beta \mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}(\mathbf{y} - \hat{\mathbf{y}})\right)^2\right] \\ &= \alpha \mathbb{E}\big[\|\mathbf{y} - \hat{\mathbf{y}}\|^2\big] + \beta d^2\, \mathbb{E}\left[\left(\frac{\frac{\partial \ell}{\partial \mathbf{y}}(\mathbf{y} - \hat{\mathbf{y}})}{d}\right)^2\right] \\ &= \alpha \mathbb{E}\big[\|\mathbf{y} - \hat{\mathbf{y}}\|^2\big] + \beta d^2\, \mathbb{E}\left[\left(\frac{\sum_{i=1}^{d}\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)_i (\mathbf{y} - \hat{\mathbf{y}})_i}{d}\right)^2\right] \\ &\leq \alpha \mathbb{E}\big[\|\mathbf{y} - \hat{\mathbf{y}}\|^2\big] + \beta d^2\, \mathbb{E}\left[\left(\frac{\sum_{i=1}^{d}\left|\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)_i (\mathbf{y} - \hat{\mathbf{y}})_i\right|}{d}\right)^2\right] \end{aligned}$$

Here, the subscript i denotes the i-th element of a vector; for instance, $(\mathbf{y} - \hat{\mathbf{y}})_i$ refers to the i-th element of $(\mathbf{y} - \hat{\mathbf{y}})$, and y_i and $\hat{y}_i$ are the i-th elements of $\mathbf{y}$ and $\hat{\mathbf{y}}$, respectively. The second term corresponds to the square of the arithmetic mean of d subterms, which can be upper bounded by the square of their quadratic mean. By applying the QM–AM inequality, we obtain:

[1] First-order Taylor approximations of the loss are widely used to estimate the effect of small parameter or structural perturbations in neural networks, due to their simplicity and empirical performance. This formulation underlies many modern sensitivity-based compression and importance estimation methods (Molchanov et al., 2017, 2019; Lee et al., 2019), and our approach follows this established first-order practice.

$$
\begin{aligned}
f(\{\mathbf{u}_k\}) &\leq \alpha\mathbb{E}\big[\|\mathbf{y}-\hat{\mathbf{y}}\|^2\big] + \beta d^2\,\mathbb{E}\left[\frac{\sum_{i=1}^{d}\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)_i^2 (\mathbf{y}-\hat{\mathbf{y}})_i^2}{d}\right] \\
&= \alpha\mathbb{E}\big[\|\mathbf{y}-\hat{\mathbf{y}}\|^2\big] + \beta d\,\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2 (\mathbf{y}-\hat{\mathbf{y}})^2\right] \\
&\approx \alpha\mathbb{E}\big[\|\mathbf{y}-\hat{\mathbf{y}}\|^2\big] + \beta d\,\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]\mathbb{E}\big[(\mathbf{y}-\hat{\mathbf{y}})^2\big] \\
&= \alpha\mathbf{1}_d^\top\mathbb{E}\big[(\mathbf{y}-\hat{\mathbf{y}})^2\big] + \beta d\,\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]\mathbb{E}\big[(\mathbf{y}-\hat{\mathbf{y}})^2\big] \\
&= \left(\alpha\mathbf{1}_d^\top + \beta d\,\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]\right)\mathbb{E}\big[(\mathbf{y}-\hat{\mathbf{y}})^2\big] \\
&= \mathbb{E}\left[\left(\alpha + \beta d\,\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]\right)(\mathbf{y}-\hat{\mathbf{y}})^2\right] \\
&= \mathbb{E}\left[\left(\sqrt{\beta d\,\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right] + \alpha}\right)^2 (\mathbf{y}-\hat{\mathbf{y}})^2\right] \\
&= \mathbb{E}\left[\sum_{i=1}^{d}\left(\left(\sqrt{\beta d\,\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right] + \alpha}\right)_i (\mathbf{y}-\hat{\mathbf{y}})_i\right)^2\right]
\end{aligned}
$$

Finally, using the Hadamard product (Definition 2), we get:

$$
f(\{\mathbf{u}_k\}) \leq \mathbb{E}\left[\left\|\sqrt{\beta d\,\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]^\top + \alpha} \odot (\mathbf{y}-\hat{\mathbf{y}})\right\|^2\right]
$$

□

A.3 Activation Space Transformation Theorem

Theorem 2. *Applying the projection condition*

$$
\mathbf{a}\odot(\hat{\mathbf{y}}-\mathbb{E}[\mathbf{y}]) = \left(\sum_k \mathbf{u}_k\mathbf{u}_k^\top\right)(\mathbf{a}\odot(\mathbf{y}-\mathbb{E}[\mathbf{y}]))
$$

where the transformation coefficient $\mathbf{a}$ *is*

$$
\mathbf{a} = \sqrt{(1-\eta)\frac{\mathbb{E}\left[\left(\frac{\partial \ell}{\partial \mathbf{y}}\right)^2\right]^\top}{\frac{1}{d}\mathbb{E}\left[\left\|\frac{\partial \ell}{\partial \mathbf{y}}\right\|^2\right]} + \eta}
$$

and utilizing the activation transformation $\tilde{\mathbf{y}} = \mathbf{a}\odot(\mathbf{y}-\mathbb{E}[\mathbf{y}])$*, the objective*

$$
h(\{\mathbf{u}_k\}) = \mathbb{E}\left[\left\|\sqrt{\frac{(1-\eta)}{\frac{1}{d}\mathbb{E}\left[\left\|\frac{\partial \ell}{\partial \mathbf{y}}\right\|^2\right]}\mathbb{E}\left[\left(\frac{\partial \ell}{\partial y}\right)^2\right]^\top + \eta} \odot (\mathbf{y}-\hat{\mathbf{y}})\right\|^2\right]
$$

becomes:

$$
h(\{\mathbf{u}_k\}) = \mathbb{E}\left[\tilde{\mathbf{y}}^\top\left(\mathbf{I}-\sum_k \mathbf{u}_k\mathbf{u}_k^\top\right)\tilde{\mathbf{y}}\right] \tag{15}
$$

Proof. Using the transformation coefficient $\mathbf{a}$, the upper bound function can be written as

$$
h(\{\mathbf{u}_k\}) = \mathbb{E}\big[\|\mathbf{a}\odot(\mathbf{y}-\hat{\mathbf{y}})\|^2\big] \tag{16}
$$

Given the projection condition

$$
\mathbf{a}\odot(\hat{\mathbf{y}}-\mathbb{E}[\mathbf{y}]) = \left(\sum_k \mathbf{u}_k\mathbf{u}_k^\top\right)(\mathbf{a}\odot(\mathbf{y}-\mathbb{E}[\mathbf{y}]))
$$

subtracting $\mathbf{a}\odot(\mathbf{y}-\mathbb{E}[\mathbf{y}])$ on both sides and, after that, multiplying both sides with -1, we have

$$
\mathbf{a}\odot(\mathbf{y}-\hat{\mathbf{y}}) = \left(\mathbf{I}-\sum_k \mathbf{u}_k\mathbf{u}_k^\top\right)(\mathbf{a}\odot(\mathbf{y}-\mathbb{E}[\mathbf{y}])) \tag{17}
$$

Combining Equations (16) and (17), we obtain:

$$
h(\{\mathbf{u}_k\}) = \mathbb{E}\left[\left\|\left(\mathbf{I}-\sum_k \mathbf{u}_k\mathbf{u}_k^\top\right)(\mathbf{a}\odot(\mathbf{y}-\mathbb{E}[\mathbf{y}]))\right\|^2\right]
$$

Given the transformed activation $\tilde{\mathbf{y}} = \mathbf{a}\odot(\mathbf{y}-\mathbb{E}[\mathbf{y}])$, the objective function can be rewritten as:

$$
\begin{aligned}
h(\{\mathbf{u}_k\}) &= \mathbb{E}\left[\left\|\left(\mathbf{I}-\sum_k \mathbf{u}_k\mathbf{u}_k^\top\right)\tilde{\mathbf{y}}\right\|^2\right] \\
&= \mathbb{E}\left[\tilde{\mathbf{y}}^\top\left(\mathbf{I}-\sum_k \mathbf{u}_k\mathbf{u}_k^\top\right)\cdot\left(\mathbf{I}-\sum_k \mathbf{u}_k\mathbf{u}_k^\top\right)\tilde{\mathbf{y}}\right] \\
&= \mathbb{E}\left[\tilde{\mathbf{y}}^\top\left(\mathbf{I}-2\sum_k \mathbf{u}_k\mathbf{u}_k^\top + \sum_{i,j}\mathbf{u}_i\mathbf{u}_i^\top\mathbf{u}_j\mathbf{u}_j^\top\right)\tilde{\mathbf{y}}\right]
\end{aligned}
$$

As $\{\mathbf{u}_k\}$ are orthogonal vectors where $\mathbf{u}_i\mathbf{u}_j^\top = 0$ if $i \neq j$, we obtain:

$$
\begin{aligned}
h(\{\mathbf{u}_k\}) &= \mathbb{E}\left[\tilde{\mathbf{y}}^\top\left(\mathbf{I}-2\sum_k \mathbf{u}_k\mathbf{u}_k^\top + \sum_k \mathbf{u}_k\mathbf{u}_k^\top\right)\tilde{\mathbf{y}}\right] \\
&= \mathbb{E}\left[\tilde{\mathbf{y}}^\top\left(\mathbf{I}-\sum_k \mathbf{u}_k\mathbf{u}_k^\top\right)\tilde{\mathbf{y}}\right]
\end{aligned}
$$

□

A.4 Weighted Covariance Matrix

Theorem 3. *The importance-weighted activation covariance matrix* $\mathbf{C}$*, given by* $\mathbf{C} = \mathbb{E}\big[\tilde{\mathbf{y}}\tilde{\mathbf{y}}^\top\big]$*, is equal to the Hadamard product of the activation covariance matrix* $\mathrm{Cov}(\mathbf{y})$ *and the gradient-informed importance matrix* $\mathbf{M}$*, i.e.,*

$$
\mathbf{C} = \mathrm{Cov}(\mathbf{y}) \odot \mathbf{M}
$$

where

$$
\mathrm{Cov}(\mathbf{y}) = \mathbb{E}\Big[(\mathbf{y}-\mathbb{E}[\mathbf{y}])(\mathbf{y}-\mathbb{E}[\mathbf{y}])^\top\Big]
$$

$$
\mathbf{M} = \mathbf{a}\mathbf{a}^\top
$$

Proof. As the importance-weighted activation covariance matrix $\mathbf{C}$ is given by $\mathbf{C} = \mathbb{E}[\tilde{\mathbf{y}}\tilde{\mathbf{y}}^\top]$, plugging the activation transformation $\tilde{\mathbf{y}} = \mathbf{a}\odot(\mathbf{y}-\mathbb{E}[\mathbf{y}])$ into this expression, matrix $\mathbf{C}$ can be written as:

$$
\mathbf{C} = \mathbb{E}\Big[(\mathbf{a}\odot(\mathbf{y}-\mathbb{E}[\mathbf{y}]))\,(\mathbf{a}\odot(\mathbf{y}-\mathbb{E}[\mathbf{y}]))^\top\Big]
$$

The $(i,j)^{\text{th}}$ element of $\mathbf{C}$ is:

$$\mathbf{C}_{ij} = \mathbb{E}[a_i(y_i - \mathbb{E}[y_i]) \cdot a_j(y_j - \mathbb{E}[y_j])]$$

where a_i, a_j and y_i and y_j are the i^{th} and j^{th} elements of $\mathbf{a}$ and $\mathbf{y}$, respectively. Since $\mathbf{a}$ is a deterministic vector, the expectation becomes:

$$\mathbf{C}_{ij} = a_i a_j \mathbb{E}[(y_i - \mathbb{E}[y_i]) \cdot (y_j - \mathbb{E}[y_j])]$$

The term $\mathbb{E}[(y_i - \mathbb{E}[y_i]) \cdot (y_j - \mathbb{E}[y_j])]$ is the $(i,j)^{\text{th}}$ element of the covariance matrix $\text{Cov}(\mathbf{y})$, which is given by:

$$\text{Cov}(\mathbf{y}) = \mathbb{E}\Big[(\mathbf{y} - \mathbb{E}[\mathbf{y}])(\mathbf{y} - \mathbb{E}[\mathbf{y}])^\top\Big]$$

Thus,

$$\mathbf{C}_{ij} = a_i a_j \text{Cov}(\mathbf{y})_{ij}$$

For the gradient-informed importance matrix $\mathbf{M}$, which is defined as $\mathbf{M} = \mathbf{a}\mathbf{a}^\top$, we have $\mathbf{M}_{ij} = a_i a_j$. Hence, the $(i,j)^{\text{th}}$ element of matrix $\mathbf{C}$ can be expressed as

$$\mathbf{C}_{ij} = \mathbf{M}_{ij} \text{Cov}(\mathbf{y})_{ij}$$

Therefore, the importance-weighted activation covariance matrix $\mathbf{C}$ is the Hadamard product of the covariance matrix $\text{Cov}(\mathbf{y})$ and the gradient-informed importance matrix $\mathbf{M}$:

$$\mathbf{C} = \text{Cov}(\mathbf{y}) \odot \mathbf{M}$$

□

Corollary 1. *The importance-weighted activation covariance matrix $\mathbf{C}$ is positive semidefinite and symmetric.*

Proof. The gradient-informed importance matrix $\mathbf{M}$, which is given by $\mathbf{M} = \mathbf{a}\mathbf{a}^\top$, is positive semidefinite and symmetric. Similarly, the covariance matrix $\text{Cov}(\mathbf{y})$, which is given by

$$\text{Cov}(\mathbf{y}) = \mathbb{E}\Big[(\mathbf{y} - \mathbb{E}[\mathbf{y}])(\mathbf{y} - \mathbb{E}[\mathbf{y}])^\top\Big],$$

is also positive semidefinite and symmetric. According to the Schur product theorem (Zhang, 2005), the Hadamard product of two positive semidefinite matrices is also positive semidefinite. Therefore, the importance-weighted activation covariance matrix $\mathbf{C} = \mathbf{M} \odot \text{Cov}(\mathbf{y})$ is positive semidefinite and symmetric. □

A.5 Reconstruction Direction Theorem

Theorem 4. *To minimize the objective $h(\{\mathbf{u}_k\})$ under orthonormality constraints, the optimal k^{th} reconstruction direction $\mathbf{u}_k$ is the eigenvector corresponding to the k^{th} largest eigenvalue of the importance-weighted activation covariance matrix $\mathbf{C}$.*

Proof. Taking the partial derivative of the Lagrangian function with respect to each reconstruction direction $\mathbf{u}_k$ and setting it to zero yields the optimality condition:

$$\frac{\partial L}{\partial \mathbf{u}_k} = -2\mathbf{u}_k^\top \mathbf{C} + 2\lambda_k \mathbf{u}_k^\top = 0$$

Rearranging this equation and taking the transpose of both sides, we obtain:

$$\mathbf{C}^\top \mathbf{u}_k = \mathbf{u}_k \lambda_k$$

Since the matrix $\mathbf{C}$ is symmetric (as established in Corollary 1), we have $\mathbf{C} = \mathbf{C}^\top$. By substituting this property, we arrive at:

$$\mathbf{C}\mathbf{u}_k = \lambda_k \mathbf{u}_k \tag{18}$$

From the Equation (15), we get:

$$\begin{aligned} h(\{\mathbf{u}_k\}) &= \mathbb{E}\Big[\tilde{\mathbf{y}}^\top \tilde{\mathbf{y}}\Big] - \mathbb{E}\left[\tilde{\mathbf{y}}^\top \sum_k \mathbf{u}_k \mathbf{u}_k^\top \tilde{\mathbf{y}}\right] \\ &= \mathbb{E}\Big[\tilde{\mathbf{y}}^\top \tilde{\mathbf{y}}\Big] - \sum_k \mathbb{E}\Big[\tilde{\mathbf{y}}^\top \mathbf{u}_k \mathbf{u}_k^\top \tilde{\mathbf{y}}\Big] \\ &= \mathbb{E}\Big[\tilde{\mathbf{y}}^\top \tilde{\mathbf{y}}\Big] - \sum_k \mathbb{E}\Big[\mathbf{u}_k^\top \tilde{\mathbf{y}} \tilde{\mathbf{y}}^\top \mathbf{u}_k\Big] \\ &= \mathbb{E}\Big[\tilde{\mathbf{y}}^\top \tilde{\mathbf{y}}\Big] - \sum_k \mathbf{u}_k^\top \mathbb{E}\Big[\tilde{\mathbf{y}} \tilde{\mathbf{y}}^\top\Big] \mathbf{u}_k \end{aligned}$$

As $\mathbf{C} = \mathbb{E}[\tilde{\mathbf{y}} \tilde{\mathbf{y}}^\top]$,

$$h(\{\mathbf{u}_k\}) = \mathbb{E}\Big[\tilde{\mathbf{y}}^\top \tilde{\mathbf{y}}\Big] - \sum_k \mathbf{u}_k^\top \mathbf{C} \mathbf{u}_k$$

Further based on Equation (18),

$$\begin{aligned} h(\{\mathbf{u}_k\}) &= \mathbb{E}\Big[\tilde{\mathbf{y}}^\top \tilde{\mathbf{y}}\Big] - \sum_k \mathbf{u}_k^\top \mathbf{u}_k \lambda_k \\ &= \mathbb{E}\Big[\tilde{\mathbf{y}}^\top \tilde{\mathbf{y}}\Big] - \sum_k \|\mathbf{u}_k\|^2 \lambda_k \end{aligned}$$

As $\|\mathbf{u}_k\|^2 = 1$,

$$h(\{\mathbf{u}_k\}) = \mathbb{E}\Big[\tilde{\mathbf{y}}^\top \tilde{\mathbf{y}}\Big] - \sum_k \lambda_k$$

To minimize $h(\{\mathbf{u}_k\})$, the term $\sum_k \lambda_k$ must be maximized. Since the importance-weighted activation covariance matrix $\mathbf{C}$ is symmetric and positive semidefinite, its eigenvalues are real and non-negative. Therefore, the maximum value of $\sum_k \lambda_k$ is achieved when λ_k is the k^{th} largest eigenvalue of $\mathbf{C}$, with $\mathbf{u}_k$ its corresponding eigenvector. □

A.6 Activation Reconstruction Theorem

Theorem 5. *The reconstructed activation $\hat{\mathbf{y}}$, which satisfies the projection condition*

$$\mathbf{a} \odot (\hat{\mathbf{y}} - \mathbb{E}[\mathbf{y}]) = \left(\sum_k \mathbf{u}_k \mathbf{u}_k^\top \right) (\mathbf{a} \odot (\mathbf{y} - \mathbb{E}[\mathbf{y}])),$$

is given by:

$$\hat{\mathbf{y}} = \left[\mathbf{U} \oslash (\mathbf{a} \cdot \mathbf{1}_r^\top) \right] \left[(\mathbf{U} \odot (\mathbf{a} \cdot \mathbf{1}_r^\top))^\top \mathbf{W} \right] \mathbf{x} + \mathbb{E}[\mathbf{y}] + (\mathbf{U}\mathbf{U}^\top \odot (\frac{1}{\mathbf{a}} \cdot \mathbf{a}^\top))(\mathbf{b} - \mathbb{E}[\mathbf{y}])$$

where $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_r]$, $\mathbf{1}_r$ is an r-dimensional column vector of ones, $\mathbf{W}$ and $\mathbf{b}$ are the original layer's weight matrix and bias, $\mathbf{x}$ is the input activation, and $\oslash$ denotes element-wise (Hadamard) division.

Proof. Rearranging the projection condition

$$\mathbf{a} \odot (\hat{\mathbf{y}} - \mathbb{E}[\mathbf{y}]) = \left(\sum_k \mathbf{u}_k \mathbf{u}_k^\top \right) (\mathbf{a} \odot (\mathbf{y} - \mathbb{E}[\mathbf{y}])),$$

we obtain

$$\begin{aligned} \mathbf{a} \odot (\hat{\mathbf{y}} - \mathbb{E}[\mathbf{y}]) &= \sum_k \mathbf{u}_k \mathbf{u}_k^\top (\mathbf{a} \odot (\mathbf{y} - \mathbb{E}[\mathbf{y}])) \\ &= \sum_k \mathbf{u}_k (\mathbf{u}_k \odot \mathbf{a})^\top (\mathbf{y} - \mathbb{E}[\mathbf{y}])) \end{aligned}$$

Applying the Hadamard division $\oslash \mathbf{a}$ to both sides of the equation leads to:

$$\hat{\mathbf{y}} - \mathbb{E}[\mathbf{y}] = \sum_k \mathbf{u}_k (\mathbf{u}_k \odot \mathbf{a})^\top (\mathbf{y} - \mathbb{E}[\mathbf{y}]) \oslash \mathbf{a}$$

Since $(\mathbf{u}_k \odot \mathbf{a})^\top$ is a row vector and $(\mathbf{y} - \mathbb{E}[\mathbf{y}])$ is a column vector, $(\mathbf{u}_k \odot a)^\top (\mathbf{y} - \mathbb{E}[\mathbf{y}])$ is a scalar, we have

$$\hat{\mathbf{y}} = \mathbb{E}[\mathbf{y}] + \sum_k (\mathbf{u}_k \oslash a)(\mathbf{u}_k \odot a)^\top (\mathbf{y} - \mathbb{E}[\mathbf{y}])$$

Rewriting the equality, we obtain:

$$\begin{aligned} \hat{\mathbf{y}} &= [\mathbf{u}_1 \oslash \mathbf{a}, \ldots, \mathbf{u}_r \oslash \mathbf{a}] \, [\mathbf{u}_1 \odot \mathbf{a}, \ldots, \mathbf{u}_r \odot \mathbf{a}]^\top (\mathbf{y} - \mathbb{E}[\mathbf{y}]) \\ &\quad + \mathbb{E}[\mathbf{y}] \\ &= \mathbb{E}[\mathbf{y}] + (\mathbf{U} \oslash (\mathbf{a} \cdot \mathbf{1}_r^\top))(\mathbf{U} \odot (\mathbf{a} \cdot \mathbf{1}_r^\top))^\top (\mathbf{y} - \mathbb{E}[\mathbf{y}]) \end{aligned}$$

where $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_r]$. Incorporating the original activation $\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b}$, we obtain:

$$\hat{\mathbf{y}} = \mathbb{E}[\mathbf{y}] + (\mathbf{U} \oslash (\mathbf{a} \cdot \mathbf{1}_r^\top))(\mathbf{U} \odot (\mathbf{a} \cdot \mathbf{1}_r^\top))^\top (\mathbf{W}\mathbf{x} + \mathbf{b} - \mathbb{E}[\mathbf{y}])$$

Expanding the expression, the reconstructed activation satisfies:

$$\begin{aligned} \hat{\mathbf{y}} &= \left[\mathbf{U} \oslash (\mathbf{a} \cdot \mathbf{1}_r^\top) \right] \left[(\mathbf{U} \odot (\mathbf{a} \cdot \mathbf{1}_r^\top))^\top \mathbf{W} \right] \mathbf{x} \\ &\quad + \mathbb{E}[\mathbf{y}] + (\mathbf{U} \oslash (\mathbf{a} \cdot \mathbf{1}_r^\top))(\mathbf{U} \odot (\mathbf{a} \cdot \mathbf{1}_r^\top))^\top (\mathbf{b} - \mathbb{E}[\mathbf{y}]) \\ &= \left[\mathbf{U} \oslash (\mathbf{a} \cdot \mathbf{1}_r^\top) \right] \left[(\mathbf{U} \odot (\mathbf{a} \cdot \mathbf{1}_r^\top))^\top \mathbf{W} \right] \mathbf{x} \\ &\quad + \mathbb{E}[\mathbf{y}] + (\mathbf{U}\mathbf{U}^\top \odot (\frac{1}{\mathbf{a}} \cdot \mathbf{a}^\top))(\mathbf{b} - \mathbb{E}[\mathbf{y}]) \end{aligned}$$

□

B Additional Results

B.1 Visualization and Analysis of the Importance Matrix

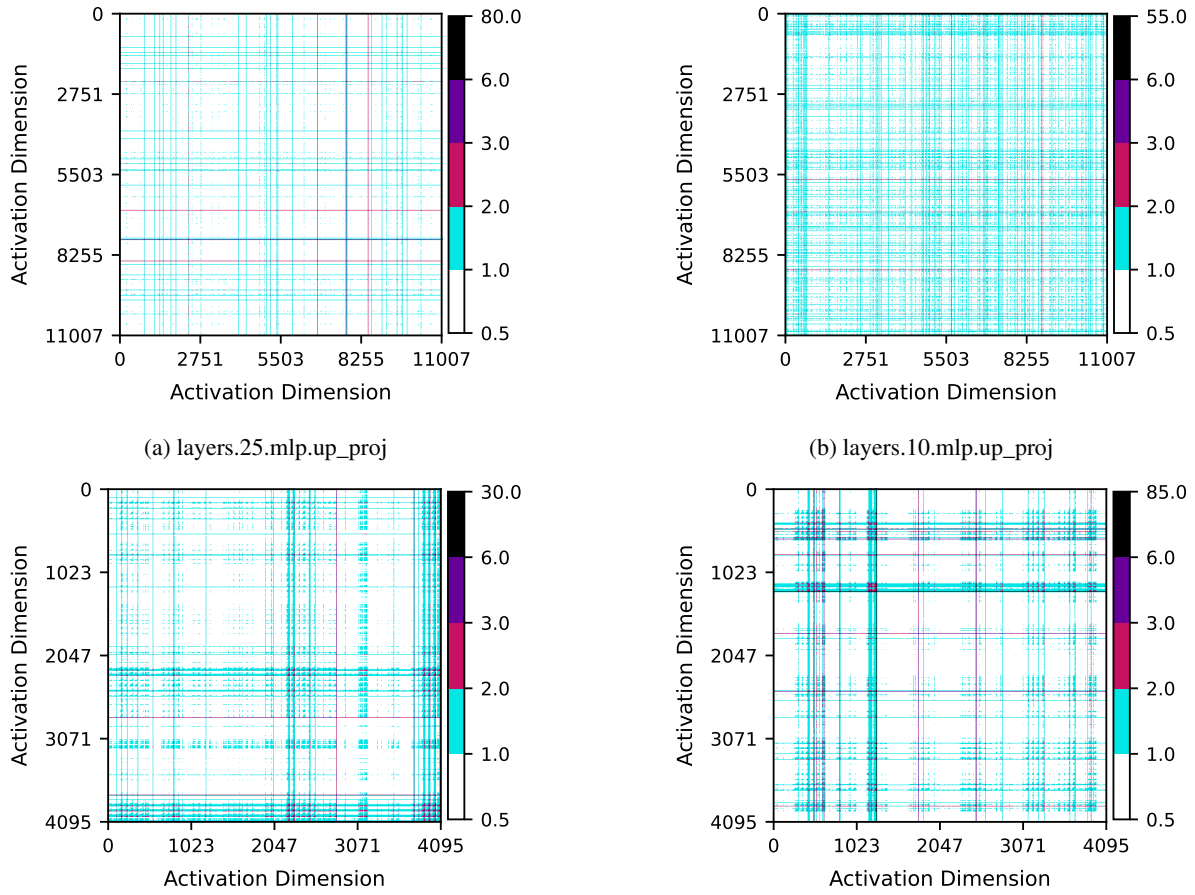
To understand how IMPACT achieves importance-aware compression, we analyze the structure and statistics of the gradient-informed importance matrix $\mathbf{M}$, offering empirical insight into its role in preserving performance-critical components.

We visualize the structure of $\mathbf{M}$ in four representative layers—two MLP and two attention—using the heatmaps in Figure 9, where darker entries indicate greater importance for interactions between activation dimensions. Across layers, $\mathbf{M}$ displays concentrated regions of high intensity, revealing that IMPACT emphasizes a sparse subset of directions rather than treating all dimensions equally. This selective weighting is central to IMPACT's design: by constructing the importance-weighted activation covariance matrix $\mathbf{C} = \mathrm{Cov}(\mathbf{y}) \odot \mathbf{M}$, the resulting eigendecomposition prioritizes performance-critical directions during compression.

Complementing the structural visualization, Figure 10 presents the distribution of values in $\mathbf{M}$ for the same layers. These histograms, plotted on a log scale, reveal highly skewed distributions. Although most entries in $\mathbf{M}$ are small, a long tail of large values persists. For instance, in "layers.25.mlp.up_proj", the median of the entries is 0.77, the mean is 0.89, the 99th percentile reaches 2.95, and the maximum is close to 80. These values quantify the relative importance of preserving interactions between activation dimensions, as determined by their gradient sensitivity during profiling. The long-tailed distribution indicates that IMPACT selectively amplifies a small subset of critical interactions. This pattern is consistent across layers and mirrors the sparsity observed in the heatmaps, reinforcing the model's emphasis on performance-relevant components.

B.2 Ablation Study on η

We analyze the effect of the hyperparameter η, which balances the reconstruction error and gradient-informed importance weighting in Equation (3). Table 4 reports results on Llama 2-7B across different compression ratios. We vary $\eta \in \{0, 0.25, 0.5, 0.75\}$ and observe that performance remains relatively stable across this range. In particular, moderate values of η generally achieve strong results across both GSM8K and MATH.



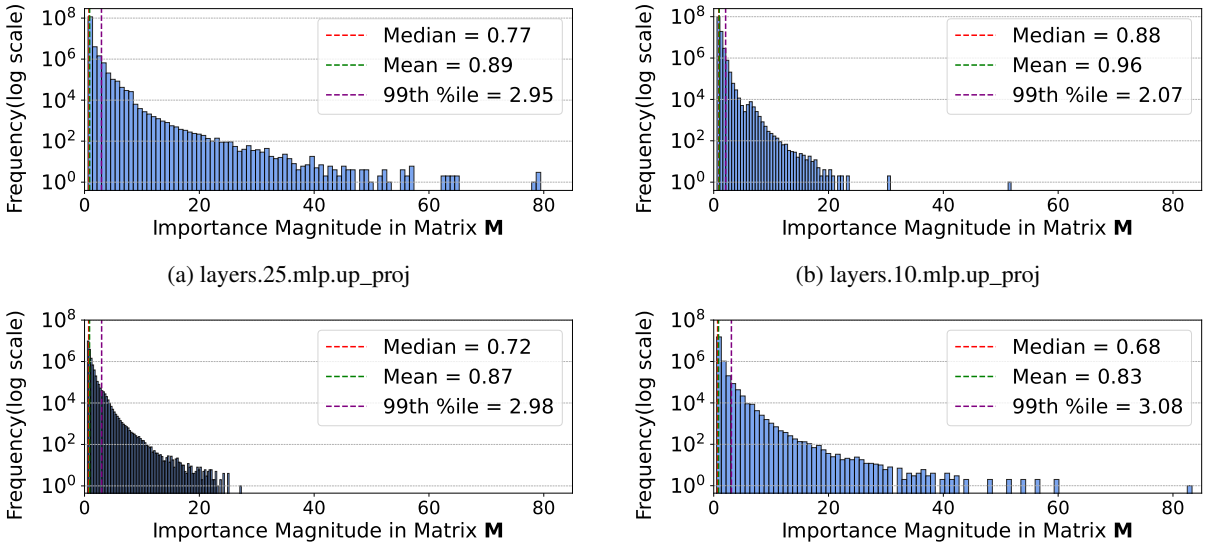
(a) layers.25.mlp.up_proj

(b) layers.10.mlp.up_proj

(c) layers.16.self_attn.q_proj

(d) layers.26.self_attn.q_proj

Figure 9: Heatmaps of the importance matrix $\mathbf{M}$ for four representative layers in Llama 2-7B on the mathematical reasoning task (with $\eta = 0.5$). High-intensity rows and columns indicate rare yet important activation dimensions that IMPACT identifies via gradient-based weighting and prioritizes during compression.



(a) layers.25.mlp.up_proj

(b) layers.10.mlp.up_proj

(c) layers.16.self_attn.q_proj

(d) layers.26.self_attn.q_proj

Figure 10: Log-scale histograms of values in the importance matrix $\mathbf{M}$ (with $\eta = 0.5$) for four representative layers of Llama 2-7B on the mathematical reasoning task.

k	Metric	$\eta = 0.75$	$\eta = 0.5$	$\eta = 0.25$	$\eta = 0$
50%	GSM8K Acc (%)	61.3	61.2	62.9	58.4
	MATH Acc (%)	17.9	17.7	17.3	15.5
60%	GSM8K Acc (%)	64.8	63.7	64.5	62.5
	MATH Acc (%)	19.2	18.4	19.4	18.1
70%	GSM8K Acc (%)	65.0	63.7	63.5	65.3
	MATH Acc (%)	18.9	19.4	18.6	19.7

Table 4: Ablation study on the hyperparameter η under different keeping ratios k. Results are reported on Llama 2-7B using GSM8K and MATH.

Method	Time (Hours)	Peak GPU Memory (GB)
IMPACT	1.53	66.9
AFM	1.17	60.8
ASVD	135.83	28.6

Table 5: Profiling cost comparison for Llama 2-13B across compression methods. All the methods are run on a single NVIDIA A100 GPU.

B.3 Profiling and Generalizability

Profiling. We note that the entire profiling process fits comfortably on a single GPU, requiring no multi-GPU setup. Using only 1% of MetaMathQA (Yu et al., 2024) and 3% of Code-Instructions-120K-Alpaca (Bisht, 2023) as calibration data, profiling completes in under two hours on a single NVIDIA A100 GPU for 13B models. Table 5 compares the profiling cost of IMPACT and baseline methods for compressing Llama 2-13B on the mathematical reasoning task. As shown in the table, IMPACT incurs only a modest additional overhead compared to AFM. This increase in profiling cost arises from the extra backward pass required to compute gradient-based importance statistics, which slightly increases both runtime and peak memory usage. In contrast, ASVD requires significantly longer profiling time. This is because ASVD performs a binary search to determine the truncation rank for each layer, repeatedly evaluating model performance for different candidate ranks. As a result, the procedure requires a substantially larger number of forward passes, leading to orders-of-magnitude longer profiling time despite lower peak GPU memory usage.

We analyze the method's sensitivity to calibration data size. We find that using only 1% of MetaMathQA or 3% of Code-Instructions-120K is sufficient for convergence. IMPACT is intentionally distribution-dependent: the profiling statistics are expectations over the calibration data, designed to maximize performance on the target domain.

Generalizability. IMPACT makes no model-specific assumptions. The closed-form solution depends strictly on activation covariance and gradient-based importance signals that are generic to all Transformer architectures. As such, the framework is inherently applicable to diverse modern LLMs.

SVD	Model Size (Billion)	6.74	5.03	3.18	1.73	1.11	0.56
	GSM8K Acc (%)	66.4	63.0	61.0	53.9	32.9	11.9
	MATH Acc (%)	20.6	18.3	17.4	13.7	5.3	2.8
FWSVD	Model Size (Billion)	6.74	4.79	2.95	1.54	0.96	0.47
	GSM8K Acc (%)	66.4	62.7	62.5	56.5	1.5	1.9
	MATH Acc (%)	20.6	19.2	17.6	14.2	1.8	1.5
ASVD	Model Size (Billion)	6.74	4.09	3.43	2.77	1.45	0.79
	GSM8K Acc (%)	66.4	61.9	60.9	59.7	50.5	29.1
	MATH Acc (%)	20.6	18.7	18. 7	17.3	12.6	5.2
AFM	Model Size (Billion)	6.74	3.62	2.66	1.90	1.30	0.83
	GSM8K Acc (%)	66.4	62.8	62.6	61.5	57.8	49.7
	MATH Acc (%)	20.6	19.5	18.5	17.6	16.3	12.0
IMPACT	Model Size (Billion)	6.74	3.11	2.40	1.67	1.19	0.75
	GSM8K Acc (%)	66.4	65.3	64.5	62.9	60.2	52.8
	MATH Acc (%)	20.6	19.7	19.4	17.3	17.2	14.1

Table 6: Pass@1 accuracy and model size of Llama 2-7B compressed by various algorithms for the mathematical reasoning task. The results for SVD and FWSVD are taken from Li et al. (2025a).

SVD	Model Size (Billion)	13.02	9.70	6.10	3.27	2.07	1.01
	GSM8K Acc (%)	72.7	69.5	63.5	50.0	26.9	6.7
	MATH Acc (%)	22.2	20.8	17.8	10.8	5.2	2.2
FWSVD	Model Size (Billion)	13.02	9.24	5.67	2.93	1.79	0.83
	GSM8K Acc (%)	72.7	67.9	63.9	51.9	2.4	3.9
	MATH Acc (%)	22.2	20.3	18.0	12.4	1.2	1.9
ASVD	Model Size (Billion)	13.02	7.87	6.56	4.03	2.73	1.45
	GSM8K Acc (%)	72.7	63.6	61.0	56.2	50.6	25.9
	MATH Acc (%)	22.2	17.7	17.4	14.0	12.0	5.6
AFM	Model Size (Billion)	13.02	9.63	5.33	3.81	2.59	1.63
	GSM8K Acc (%)	72.7	70.9	66.4	65.7	60.8	47.2
	MATH Acc (%)	22.2	20.9	18.5	18.0	16.3	13.3
IMPACT	Model Size (Billion)	13.02	8.66	4.87	3.58	2.28	1.49
	GSM8K Acc (%)	72.7	70.2	69.3	65.3	61.2	54.8
	MATH Acc (%)	22.2	21.2	19.4	18.6	17.7	15.2

Table 7: Pass@1 accuracy and model size of Llama 2-13B compressed by various algorithms for the mathematical reasoning task. The results for SVD and FWSVD are taken from Li et al. (2025a).

B.4 Additional Tables

Tables 6–9 report the precise Pass@1 accuracy and compressed model sizes corresponding to the performance curves plotted in Figures 2–5. They cover both Llama 2 (mathematical reasoning) and CodeLlama (code generation) across 7B and 13B.

To quantify compression gains, we compare IMPACT with the smallest baseline model that attains equivalent accuracy. Detailed size reductions are reported in Tables 10 and 11 for Llama 2-7B and -13B, respectively, and in Tables 12 and 13 for

CodeLlama-7B and -13B, respectively. Table 14 reports the inference throughput and memory consumption of the compressed models.

SVD	Model Size (Billion)	6.74	6.14	3.13	2.37	1.69	1.09
	HumanEval Acc (%)	45.1	37.8	27.4	14.0	9.8	3.0
	MBPP Acc (%)	59.8	54.8	46.8	35.7	19.6	6.1
FWSVD	Model Size (Billion)	6.74	4.77	2.94	2.19	1.54	0.96
	HumanEval Acc (%)	45.1	28.7	22.6	20.1	9.1	3.7
	MBPP Acc (%)	59.8	54.8	44.2	36.2	23.8	9.5
ASVD	Model Size (Billion)	6.74	4.10	3.43	2.77	2.11	1.45
	HumanEval Acc (%)	45.1	27.4	26.8	22.6	14.6	5.5
	MBPP Acc (%)	59.8	51.9	44.7	43.9	32.3	16.1
AFM	Model Size (Billion)	6.74	3.76	2.81	2.04	1.42	0.92
	HumanEval Acc (%)	45.1	31.7	25.6	19.5	4.9	4.9
	MBPP Acc (%)	59.8	49.5	49.2	43.1	26.7	12.2
IMPACT	Model Size (Billion)	6.74	3.58	2.64	1.90	1.30	0.84
	HumanEval Acc (%)	45.1	33.5	27.4	23.2	11.0	4.9
	MBPP Acc (%)	59.8	51.6	49.7	42.6	28.0	16.4

Table 8: Pass@1 accuracy and model size of CodeLlama-7B compressed with various low-rank algorithms on the code generation task. The results for SVD and FWSVD are taken from Li et al. (2025a).

SVD	Model Size (Billion)	13.02	9.54	5.94	4.47	3.16	1.77
	HumanEval Acc (%)	52.4	36.6	28.0	18.3	14.6	1.8
	MBPP Acc (%)	63.0	58.2	49.7	47.4	30.7	1.3
FWSVD	Model Size (Billion)	13.02	9.17	5.29	4.14	2.30	1.76
	HumanEval Acc (%)	52.4	38.4	29.3	21.3	13.4	2.4
	MBPP Acc (%)	63.0	57.7	46.6	44.7	23.0	4.5
ASVD	Model Size (Billion)	13.02	9.16	5.32	4.01	2.73	1.45
	HumanEval Acc (%)	52.4	28.7	19.5	20.1	6.1	4.3
	MBPP Acc (%)	63.0	53.4	44.2	38.9	29.4	10.3
AFM	Model Size (Billion)	13.02	9.68	5.45	3.96	2.74	1.75
	HumanEval Acc (%)	52.4	43.3	34.8	21.3	14.6	6.1
	MBPP Acc (%)	63.0	59.8	51.3	43.4	31.2	19.0
IMPACT	Model Size (Billion)	13.02	9.61	5.16	3.71	2.44	1.66
	HumanEval Acc (%)	52.4	47.6	34.1	20.7	15.2	8.5
	MBPP Acc (%)	63.0	61.6	51.3	45.5	31.7	25.9

Table 9: Pass@1 accuracy and model size of CodeLlama-13B compressed with various low-rank algorithms on the code generation task. The results for SVD and FWSVD are taken from Li et al. (2025a).

GSM8K							
Best Baseline[2] (with Same Acc.)	Model Size (Billion)	6.74	5.76	5.12	3.75	1.69	1.02
IMPACT	Model Size (Billion)	6.74	3.11	2.40	1.67	1.19	0.75
	Size Reduction (%)	-	46.0	53.2	55.4	29.8	25.7
MATH							
Best Baseline[2] (with Same Acc.)	Model Size (Billion)	6.74	4.15	3.51	1.77	1.71	1.06
IMPACT	Model Size (Billion)	6.74	3.11	2.40	1.67	1.19	0.75
	Size Reduction (%)	-	25.0	31.7	5.6	30.5	28.7

Table 10: Size reduction of IMPACT compared to the best baseline at matched accuracy for Llama 2-7B on the mathematical reasoning task.

GSM8K							
Best Baseline[2] (with Same Acc.)	Model Size (Billion)	13.02	8.97	8.10	3.70	2.69	2.17
IMPACT	Model Size (Billion)	13.02	8.66	4.87	3.58	2.28	1.49
	Size Reduction (%)	-	3.5	39.8	3.3	15.0	31.2
MATH							
Best Baseline[2] (with Same Acc.)	Model Size (Billion)	13.02	10.33	6.91	5.44	3.59	2.25
IMPACT	Model Size (Billion)	13.02	8.66	4.87	3.58	2.28	1.49
	Size Reduction (%)	-	16.2	29.4	34.2	36.5	33.7

Table 11: Size reduction of IMPACT compared to the best baseline at matched accuracy for Llama 2-13B on the mathematical reasoning task.

HumanEval							
Best Baseline[2] (with Same Acc.)	Model Size (Billion)	6.74	4.16	3.09	2.51	1.65	1.09
IMPACT	Model Size (Billion)	6.74	3.58	2.64	1.90	1.30	0.84
	Size Reduction (%)	-	13.8	14.4	24.2	21.0	23.1
MBPP							
Best Baseline[2] (with Same Acc.)	Model Size (Billion)	6.74	4.07	3.82	2.02	1.47	1.06
IMPACT	Model Size (Billion)	6.74	3.58	2.64	1.90	1.30	0.84
	Size Reduction (%)	-	12.0	30.7	6.1	11.1	21.1

Table 12: Size reduction of IMPACT compared to the best baseline at matched accuracy for CodeLlama-7B on the code generation task.

HumanEval							
Best Baseline[2] (with Same Acc.)	Model Size (Billion)	13.02	11.25	5.38	3.85	2.72	2.03
IMPACT	Model Size (Billion)	13.02	9.61	5.16	3.71	2.44	1.66
	Size Reduction (%)	-	14.6	3.9	3.7	10.2	18.1
MBPP							
Best Baseline[2] (with Same Acc.)	Model Size (Billion)	13.02	11.55	5.45	4.32	2.79	2.31
IMPACT	Model Size (Billion)	13.02	9.61	5.16	3.71	2.44	1.66
	Size Reduction (%)	-	16.9	5.3	14.2	12.5	28.0

Table 13: Size reduction of IMPACT compared to the best baseline at matched accuracy for CodeLlama-13B on the code generation task.

SVD	Model Size (Billion)	6.74	5.03	3.18	1.73	1.11	0.56
	Throughput (Token/s)	261.74	321.71	373.47	489.00	545.19	607.92
	Memory (GB)	29.26	22.73	15.34	9.75	6.90	4.70
FWSVD	Model Size (Billion)	6.74	4.79	2.95	1.54	0.96	0.47
	Throughput (Token/s)	261.74	341.30	402.54	506.01	553.89	639.70
	Memory (GB)	29.26	21.93	14.56	9.04	6.62	4.38
AFM	Model Size (Billion)	6.74	3.62	2.66	1.90	1.30	0.83
	Throughput (Token/s)	261.74	344.59	414.04	470.98	538.42	566.43
	Memory (GB)	29.26	17.03	13.24	10.21	7.70	5.76
IMPACT	Model Size (Billion)	6.74	3.11	2.40	1.67	1.19	0.75
	Throughput (Token/s)	261.74	414.03	445.16	505.74	568.04	616.01
	Memory (GB)	29.26	14.99	12.21	9.25	7.25	5.41

Table 14: Throughput and memory consumption of compressed models. The results for SVD and FWSVD are taken from Li et al. (2025a).

[2]The best baseline refers to the smallest model among SVD, FWSVD, ASVD, and AFM that achieves accuracy matched to IMPACT. If no baseline exactly matches the accuracy, the model size is interpolated linearly between two adjacent compression points.