Rethinking Parameter Sharing as Graph Coloring for Structured Compression

Anonymous Authors[1]

Abstract

Parameter sharing is a key model compression technique, yet existing methods overlook the geometric properties of the loss landscape, often causing severe accuracy degradation under high compression ratios. Inspired by second-order optimization, we propose Curvature-aware Graph Coloring (CGC), a cross-layer parameter sharing framework that treats each network layer as a graph node, with each node assigned to a shared low-rank basis. CGC leverages Hessian eigenspace information to group layers with similar curvature profiles, aligning the perturbations introduced by parameter sharing with the low-curvature (flat) directions of the loss ellipsoid. This effectively mitigates performance loss while enabling flexible, global cross-layer sharing. Experiments on LLaMA-7B and Swin Transformer show that CGC achieves 28%–50% parameter compression with Top-1 accuracy loss no more than 0.01% on Swin—or even accuracy gains on LLaMA—while delivering over 60% higher inference throughput, significantly outperforming SVD-based and heuristic-based methods. This work demonstrates that curvature-guided, geometry-aware sharing is essential for efficient, stable, and high-ratio model compression.

1. Introduction

Parameter sharing is pivotal in model compression. But current parameter-sharing approaches still face a fundamental performance challenge: the accuracy loss after compression is difficult to control, especially under high compression ratios where performance often degrades sharply. The root cause is that existing methods predominantly rely on rigid heuristic rules to design sharing structures—for example, enforcing sharing only between adjacent layers, partitioning weights into fixed-size blocks, or manually grouping layers by index. These strategies ignore the true distribution of parameter redundancy within the network and fail to account for the model's differential sensitivity to perturbations along different directions. Consequently, even if the introduced perturbation is small in Euclidean norm, it can still cause significant performance degradation if aligned with high-sensitivity regions of the loss landscape.

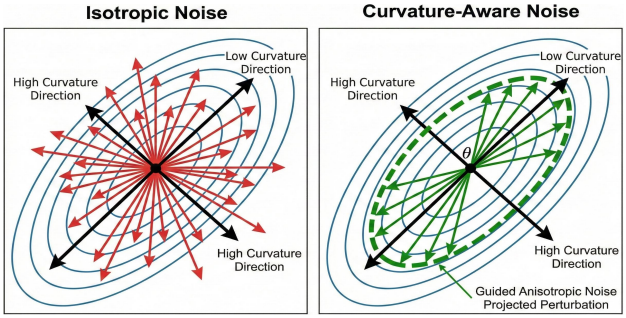

Figure 1. Geometric interpretation of sharing noise. Existing methods fail to control the perturbation direction, resulting in isotropic noise (red) that inevitably intersects high-curvature axes. In contrast, our method guides perturbations along low-curvature directions (green), confining them to "flat" regions.

Existing works (Hassibi et al., 1993; Frantar et al., 2023; Zhang et al., 2025) shows that the noise induced by parameter sharing can be effectively characterized by second-order information—specifically, the Hessian matrix. When a small perturbation is applied to the parameters, the resulting increase in loss is primarily determined by the projection of that perturbation onto the Hessian's eigenspace. Locally, the loss surface can be approximated as a multidimensional ellipsoid: its minor axes correspond to high-curvature directions (large eigenvalues), where the loss is highly sensitive to parameter changes; in contrast, its major axes correspond to low-curvature directions (small eigenvalues), representing flat regions where parameter variations have minimal impact on performance.

Therefore, an ideal compression-induced perturbation should be aligned as closely as possible with the ellipsoid's major axes. This not only maximizes tolerance to parameter changes but also ensures stable performance. As illustrated in Figure 1, while traditional methods introduce isotropic noise (red arrows) that inevitably intersects high-curvature contours, our approach confines perturbations to the low-curvature valley (green arrows), effectively "sliding" along

[1]. Correspondence to: .

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

the flat directions of the loss landscape.

Motivated by this geometric insight, we propose Curvature-aware Graph Coloring (CGC), a novel cross-layer parameter sharing framework. We treat each layer of the model as a node in a graph and formalize the design of the sharing structure as a graph coloring problem: each "color" corresponds to a shared low-rank basis, and the coloring process aims to assign layers with similar Hessian geometric properties to the same color group. Unlike traditional heuristic approaches, CGC explicitly leverages Hessian eigenspace from each layer as the matching criterion when constructing and selecting shared bases. Specifically, by analyzing the curvature profile of each layer's loss landscape, we identify its dominant low-curvature directions and enforce alignment with these directions during basis assignment. This ensures that the perturbation introduced by parameter sharing predominantly resides near the major axes of the loss ellipsoid.

This design yields distinct advantages. First, perturbations are actively steered toward directions least harmful to performance, significantly mitigating accuracy loss. Second, because the coloring process is no longer constrained by topological proximity, our method naturally supports flexible, arbitrary cross-layer sharing. This allows it to uncover deep redundancies across the entire network and transcend the limitations of adjacent-layer sharing. Furthermore, by formalizing sharing through graph theory, our framework implicitly captures the structural symmetry of the parameter, offering a rigorous mathematical grounding for redundancy reduction. Our key contributions are summarized as follows:

• **Geometric-Algebraic Framework**: We formalize cross-layer parameter sharing as a graph coloring problem. By incorporating automorphism groups to characterize structural symmetry, we provide a theoretical guarantee for capturing deep inter-layer redundancies beyond simple heuristics.

• **Curvature-aware Mechanism**: We propose a novel coloring function α that leverages Hessian analysis. This mechanism actively aligns shared parameters with low-curvature directions, avoiding the exponential complexity of combinatorial search while minimizing accuracy degradation.

• **Performance**: We validate CGC on LLaMA-7B and Swin Transformer. At 28%–50% compression rates, CGC maintains Top-1 accuracy loss within 0.01% (or achieves gains) and boosts inference throughput by over 60%, significantly outperforming SVD-, and heuristic-based sharing baselines.

2. Related Work

SVD-based Weight Decomposition. As deep models grow increasingly large, their memory and compute overhead at inference time severely limits deployment in edge scenarios, where resources are constrained. Model compression is therefore crucial for enabling practical deployment. Singular value decomposition (SVD) and low-rank approximation are widely adopted techniques for neural network compression (Golub et al., 1987; Vaswani et al., 2017; Lv et al., 2023; Wu et al., 2023; Hsu et al., 2022). There are many works to improve the performance of SVD based methods (Zhang et al., 2024; Wang et al., 2025c; Yuan et al., 2025). Despite these advances, most methods treat each layer independently, overlooking structural redundancies that could be exploited across layers—a limitation addressed by CGC.

Parameter Sharing. Parameter sharing (Wang et al., 2025b) reduces model size by reusing weights across layers. Universal Transformer (Dehghani et al., 2018) shares weights entirely across encoder and decoder layers, while Subformer (Reid et al., 2021) divides parameters into attention and feed-forward groups for intra-group sharing. Most methods adopt group-wise strategies, where identical weights are shared within predefined groups. Dynamic Tying (Hay & Wolf, 2024) explores sharing structures during training via reinforcement learning, but its computational cost limits scalability to large models. Training-free methods like FiPS (Üyük et al., 2025) minimize block-level reconstruction error for ViTs, yet restrict sharing to adjacent blocks. Basis Sharing (Wang et al., 2025a) improves this by representing adjacent layers with shared basis and coefficients. However, these methods rely on heuristics and fail to explore comprehensive sharing across multiple layers—a gap addressed by CGC, which systematically identifies shared structures spanning layers to optimize compression.

3. Approach

3.1. Curvature-aware Graph Coloring

Deep neural networks, upon convergence, exhibit a distinct geometric structure in the loss landscape: locally, the loss surface can be approximated by an ellipsoid, where the major axes correspond to low-curvature (flat) directions and the minor axes to high-curvature (sensitive) directions. This geometric insight reveals a key principle—if parameter perturbations introduced during model compression can be steered toward flat directions, significant parameter reduction can be achieved while preserving model performance.

Motivated by this observation, we propose Curvature-aware Graph Coloring (CGC), shown in Algorithm 1. The pipeline of CGC is shown in Figure 2. CGC treats each network layer as a node in a graph and formulates cross-layer parameter sharing as a graph coloring problem: each "color" represents a shared low-rank basis, and the coloring process aims to assign layers with similar curvature characteristics to the same color group. Crucially, we characterize the local geometry of each layer using the eigenspace of its Hessian matrix and measure inter-layer similarity based on their low-

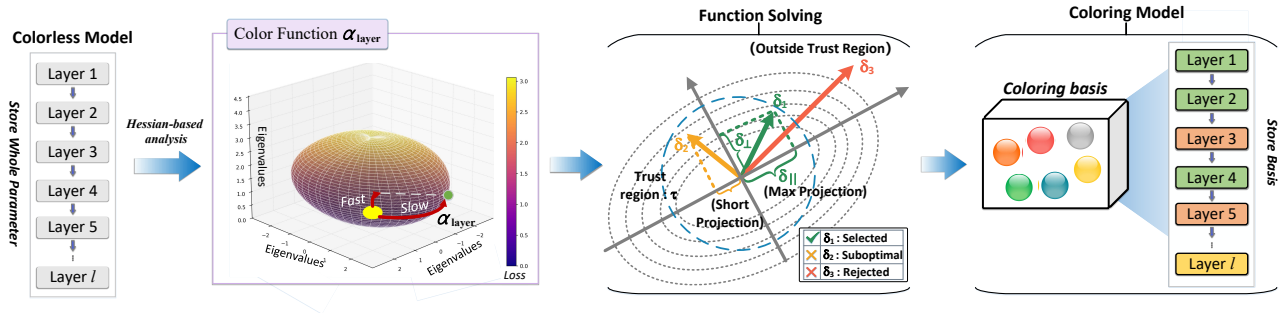

Figure 2. Curvature-aware Graph Coloring (CGC) pipeline: Starting from the original parameters (Colorless Model), we analyze the local loss geometry via Hessian analysis to identify high- and low-curvature directions. In the Function Solving stage, the algorithm selects a perturbation δ that aligns with the flat directions to minimize error, strictly constrained within a trust region τ (e.g., δ_1 is selected as optimal, while δ_3 is rejected for violating the constraint). Finally, layers assigned to the same basis are grouped in the Coloring Model.

curvature principal directions. During basis assignment, the algorithm prioritizes alignment along these flat directions, ensuring that sharing-induced perturbations primarily reside in the subspace where the loss changes most slowly. This curvature-guided matching naturally confines compression error along the ellipsoid's major axes, enabling stable and low-loss compression. Moreover, the graph coloring framework supports flexible, global grouping across the entire network—without relying on predefined adjacency or fixed patterns—significantly enhancing redundancy exploitation and compression efficiency.

3.2. Parameter Sharing as Graph Coloring

We formulate cross-layer parameter sharing as a geometry-guided graph coloring problem. Specifically, each layer of the neural network is treated as a node in a graph. An edge is implicitly established between two layers if they exhibit similar local geometric properties in the loss landscape—namely, comparable sensitivity to parameter perturbations—indicating their potential to share a common structural basis. The goal is to assign a "color" to each node, where each color corresponds to one shared basis. This global perspective enables our method to move beyond heuristic constraints such as "adjacent-layer-only" sharing and automatically discover functionally redundant structures that recur across the entire model depth.

Crucially, CGC does not enforce identical weight values across layers, which would severely limit expressivity. We therefore follow a more flexible strategy: sharing the underlying structural properties of weight matrices instead of their exact values. To achieve this, we employ low-rank decomposition. Each shared basis B_b is defined as a pair of factor matrices $B_b = (U_b, V_b)$. A layer ℓ that is assigned to this basis constructs its weight matrix W_ℓ by combining the shared basis with a small, layer-specific coefficient matrix $S_{\ell,b}$: $W_\ell \approx U_b S_{\ell,b} V_b^T$. Here, $U_b \in \mathbb{R}^{M\times r}$ and $V_b \in \mathbb{R}^{r\times N}$ are the shared factors, while $S_{\ell,b} \in \mathbb{R}^{r\times r}$ preserves layer-specific information. The rank r acts as a trade-off between compression and expressivity. The task thus becomes finding an optimal coloring function. Specifically, we treat this as an assignment problem on a complete graph. The objective is not to satisfy adjacency constraints, but, for a given set of candidate basis $\mathcal{B}$ (generated following (Wang et al., 2025a)), to find an assignment function α_{layer} that maps each layer to a basis in $\mathcal{B}$ that minimizes performance impact.

By constraining same-color layers to share identical basis factors (U, V), this approach induces structural symmetry, rendering these layers functionally interchangeable. We observe that these symmetric configurations naturally gravitate toward flatter regions of the loss landscape, even without explicit optimization. This indicates that our method effectively uncovers intrinsic redundancy to achieve robustness.

3.3. Curvature-aware Coloring Function

Problem Formulation via Loss Geometry. The core objective of our framework is to determine the optimal assignment function, α_{layer}, which maps each layer ℓ to a shared basis from a candidate set $\mathcal{B} = \{B_k\}_{k=1}^K$, where each basis B_k consists of a pair of low-rank matrices U_k, V_k. Rather than relying on heuristic weight distances, we formulate this selection based on the local geometry of the loss landscape. Let W denote the original parameters and $\widehat{W}(\alpha_{\text{layer}})$ be the shared approximation. The perturbation introduced by sharing is defined as $\delta = \widehat{W} - W$.

For a converged model, the first-order gradient vanishes. Consequently, the degradation in performance, ΔJ, is dominated by the second-order term in the Taylor expansion:

$$\Delta J \approx \frac{1}{2}\delta^\top H \delta, \tag{1}$$

where H is the Hessian matrix. Geometrically, the level sets of this quadratic form define a multidimensional ellipsoid. The eigenvectors of H determine the directions of the ellipsoid's axes. Specifically, eigenvectors associated with large

eigenvalues correspond to high-curvature directions where the loss increases rapidly; these represent the ellipsoid's *minor axes*. In contrast, eigenvectors associated with small eigenvalues correspond to low-curvature (flat) directions where the loss is insensitive to perturbations; these represent the ellipsoid's *major axes*.

Algorithm 1 Curvature-Aware Graph Coloring Algorithm

Input: Original weights $\{W_\ell\}$, candidate basis $\{B_b = (U_b, V_b)\}$, amplitude factor β, minor-axis count t.
Output: Assignment α_{layer}, Layer Coefficients $\{S_\ell\}$.

Precompute minor-axis vectors $\{p_j^{(\ell)}\}$ for each layer ℓ
for each layer ℓ **do**
  $\tau_\ell \leftarrow \beta \cdot \|W_\ell\|_F$
  // Select basis minimizing high-curvature energy, subject to total perturbation constraint
  $b^* \leftarrow \arg\min_{b\in\{1..K\}} \left\|\delta_\perp^{(b)}\right\|_2^2 \quad \text{s.t.} \quad \left\|\delta_\parallel^{(b)}\right\|_2 \le \tau_\ell$
  where
    $\delta^{(b)} = U_b(U_b^\top W_\ell V_b)V_b^\top - W_\ell$
    $\delta_\perp^{(b)} = \sum_{j=1}^{t} \langle p_j^{(\ell)}, \delta^{(b)} \rangle p_j^{(\ell)}$
  $\alpha_{\text{layer}}(\ell) \leftarrow b^*$
  $S_\ell \leftarrow U_{b^*}^\top W_\ell V_{b^*}$
end for
return $\alpha_{\text{layer}}, \{S_\ell\}$

Subspace Decomposition and Alignment. To minimize accuracy loss, the perturbation δ must be aligned with the major axes (flat directions) of the loss ellipsoid. We formally decompose the perturbation into 2 orthogonal components:

> **High-curvature component ($\delta_\perp$):** The projection of δ onto the subspace spanned by the top-t dominant eigenvectors (corresponding to the minor axes).

> **Low-curvature component ($\delta_\parallel$):** The projection onto the remaining flat subspace (corresponding to the major axes).

Since the eigenvalues in the high-curvature subspace are significantly larger, the optimization priority is to minimize the energy of $\delta_\perp$. However, we must also ensure the total perturbation remains within a reasonable range to validate the local quadratic approximation.

Optimization Objective. Based on this geometric insight, we define the coloring function as a constrained optimization problem. For each layer ℓ, we select the basis b^* that minimizes the projection error on the high-curvature directions, subject to a trust-region constraint on the total perturbation:

$$b^* = \arg\min_{b\in\{1..K\}} \left\|\delta_\perp^{(b)}\right\|_2^2 \quad \text{s.t.} \quad \left\|\delta_\parallel^{(b)}\right\|_2 \le \tau_\ell, \qquad (2)$$

where $\tau_\ell = \beta\|W_\ell\|_F$ defines the radius of the trust region, controlled by a hyperparameter β. This constraint prevents the shared parameters from drifting too far from the original weights, even along the major axes, ensuring the stability of the approximation.

3.4. Problem Solving and Parameter Reconstruction

The optimization problem from Eq. (2) is solved greedily, as detailed in Algorithm 1. This procedure operationalizes our graph-coloring framework: for each layer ℓ (a node), the algorithm iterates through every candidate basis B_b (a color, generated following (Wang et al., 2025a)). It then selects the optimal color b^* that minimizes the projection error onto the high-curvature subspace—our effective "coloring cost"—while respecting the trust-region constraint. The final output is the coloring function, α_{layer}, which maps each layer to its optimal basis. This function does more than assign parameters; it uncovers latent structural properties. All layers sharing the same color (e.g., $\ell \mid \alpha_{\text{layer}}(\ell) = b^*$) form a set of interchangeable components, and the resulting partitioning of layers defines a non-trivial subgroup of the model's automorphism group, discovered automatically by our method. If no candidate basis satisfies the trust-region constraint $\|\delta_\parallel^{(b)}\|_2 \le \tau_\ell$, we gradually relax τ_ℓ by a factor of 1.2 until feasibility is reached, which guarantees convergence within two iterations in all experiments.

Efficient Reconstruction. This procedure of determining a 'color' for each layer directly yields a compressed model. Instead of storing full-rank weight matrices, for each layer ℓ we only store its assigned basis index (color), $\alpha_{\text{layer}}(\ell)$, and a small, layer-specific coefficient matrix S_ℓ. The full weight matrix $\widehat{W}_\ell$ is then reconstructed on-the-fly only when needed during inference:

$$\widehat{W}_\ell = U_{b^*} S_\ell V_{b^*}^\top, \quad \text{where } b^* = \alpha_{\text{layer}}(\ell). \qquad (3)$$

This low-rank formulation is the source of our method's efficiency. Parameter savings arise from storing the compact matrices S_ℓ instead of the full weights W_ℓ. Simultaneously, inference is accelerated because the single, expensive full-matrix multiplication is replaced by two faster, smaller matrix multiplications.

4. Experiments

We structure the experiments into three parts: **(1)** Downstream performance across Large Language Models (LLMs) and Vision Transformers to verify the method's generalizability and fidelity retention; **(2)** CGC parameter sharing analysis, demonstrating the reduction in shared parameters and the curvature effectiveness; and **(3)** Computational benchmarks, highlighting enhanced inference efficiency.

Table 1. Our method's PPL (↓) and zero-shot (↑) performance under LLaMA-7B, following an SVD-based evaluation scheme on 3 language modeling datasets and 7 common-sense reasoning datasets(%). Ratio represents the compression rate.

Ratio	Method	PTB↓	C4↓	WikiText-2↓	Openb.	ARC_e	WinoG.	HellaS.	ARC_c	PIQA	MathQA	Average↑
0%	Original	8.35	7.34	5.68	28.0	67.0	67.0	56.0	38.0	78.0	27.0	52.0
20%	SVD	20306	18800	20061	14.0	27.0	51.0	26.0	21.0	53.0	21.0	31.0
	FWSVD	2152	1511	1727	15.0	31.0	50.0	26.0	23.0	56.0	21.0	32.0
	ASVD	16.55	15.93	11.14	25.0	53.0	64.0	41.0	27.0	68.0	24.0	43.0
	SVD-LLM	18.05	15.93	7.94	22.0	58.0	63.0	43.0	29.0	69.0	24.0	44.0
	Basis Sharing	17.35	15.03	7.74	28.0	66.0	66.0	46.0	36.0	71.0	25.0	48.0
	CGC(Ours)	**16.54**	**13.88**	**7.07**	**29.0**	**66.1**	**68.5**	**46.4**	**37.4**	**71.1**	**25.1**	**49.1**
30%	SVD	17210	20871	13103	13.0	26.0	51.0	26.0	21.0	54.0	22.0	30.0
	FWSVD	11058	7240	20127	17.0	26.0	49.0	26.0	22.0	51.0	19.0	30.0
	ASVD	70	41	51	18.0	43.0	53.0	37.0	25.0	65.0	21.0	38.0
	SVD-LLM	29.44	25.11	9.56	20.0	48.0	59.0	40.0	26.0	65.0	22.0	40.0
	Basis Sharing	29.12	22.46	9.25	27.0	63.0	63.0	40.0	30.0	68.0	24.0	45.0
	CGC(Ours)	**27.65**	**21.89**	**9.13**	**28.1**	**64.5**	**65.8**	**41.3**	**33.0**	**68.9**	**24.3**	**46.7**
40%	SVD	59977	47774	52489	15.0	26.0	52.0	26.0	22.0	53.0	20.0	30.0
	FWSVD	20990	12847	18156	16.0	26.0	51.0	26.0	22.0	53.0	21.0	30.0
	ASVD	3292	1109	1407	13.0	28.0	48.0	26.0	22.0	55.0	19.0	30.0
	SVD-LLM	63.75	49.83	13.11	19.0	42.0	58.0	33.0	25.0	60.0	21.0	37.0
	Basis Sharing	55.78	41.28	12.39	22.0	52.0	61.0	35.0	27.0	62.0	**23.0**	40.0
	CGC(Ours)	**52.47**	**39.78**	**12.16**	**23.4**	**54.9**	**62.4**	**35.6**	**28.4**	**64.3**	**23.0**	**41.8**
50%	SVD	87227	79815	131715	16.0	26.0	50.0	26.0	23.0	52.0	19.0	30.0
	FWSVD	28321	23104	24391	12.0	26.0	50.0	26.0	23.0	53.0	20.0	30.0
	ASVD	47690	27925	15358	12.0	26.0	51.0	26.0	22.0	52.0	19.0	30.0
	SVD-LLM	150.58	118.57	23.97	16.0	33.0	54.0	29.0	23.0	56.0	21.0	33.0
	Basis Sharing	126.35	88.44	19.99	18.0	42.0	57.0	**31.0**	23.0	58.0	22.0	36.0
	CGC(Ours)	**117.23**	**79.01**	**18.95**	**19.6**	**44.7**	**59.6**	**31.0**	**24.0**	**60.0**	**22.1**	**37.3**

4.1. Experiment Configuration

4.1.1. Model

Visual Transformers. For visual Transformers, we evaluate the Swin-Transformer (Liu et al., 2021) on ImageNet (Krizhevsky et al., 2012) and transfer it to downstream tasks such as CIFAR (Krizhevsky et al., 2009).

Large Language Model. For large language models (LLMs), we conduct experiments on multiple architectures, including the Qwen3 family(Qwen et al., 2025), LLaMA (Touvron et al., 2023) family (LLaMA-7B, LLaMA-13B, LLaMA-30B, LLaMA2-7B, LLaMA3.1-8B, LLaMA3.2 3B), OPT-6.7B (Zhang et al., 2022), and Mistral-7B.

4.1.2. Dataset and Configuration

Dataset. Our evaluation encompasses 3 language modeling datasets: WikiText-2 (Merity et al., 2016), Penn Treebank (PTB) (Marcinkiewicz, 1994), and C4 (Raffel et al., 2020). Additionally, we assess performance on seven reasoning datasets: OpenbookQA (Banerjee et al., 2019), WinoGrande (Sakaguchi et al., 2021), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), MathQA (Amini et al., 2019), ARC-easy, and ARC-challenge (Clark et al., 2018). All reasoning tasks are evaluated under zero-shot settings using the LM-Evaluation-Harness framework to ensure consistent and reproducible results.

Configuration. All models are implemented using Hugging Face transformers. LLaMA-30B is implemented in FP16 precision, while all other models use FP32. For sharing, we follow the Basis Sharing. Hessian terms and eigenvectors are approximated using Hessian-vector products combined with the Lanczos algorithm. The number of short-axis eigenvalues t is set to 550, and the perturbation amplitude β is set to 5e-2; these hyperparameter choices will be justified in the ablation studies. Following (Wang et al., 2025a), we construct $\mathcal{B} = \{B_k\}_{k=1}^{K}$ by aggregating layerwise weight matrices $\{W_\ell\}_{\ell \in L}$ within each parameter type (Q, K, V, FFN, etc.). Specifically, for each type we perform SVD on the concatenated matrices: $\tilde{W} = [W_1; W_2; \dots]$, retain the top-$r$ singular vectors as the shared basis B_k. The number of basis K is determined by the target compression ratio ρ. For detailed K, r, ρ, refer to **Appendix C**. Experiments were implemented in PyTorch on 2 NVIDIA A800 GPUs.

4.2. Downstream Performance Experiments

4.2.1. Large Language Model

Comparison Results. As shown in Table 1, across compression ratios, our method consistently outperforms all

Table 2. Comparison of our method's PPL (↓) performance on LLaMA2-7B with the baseline under different compression ratios.

Ratio	Method	PTB↓	C4↓	WikiText-2↓
0%	Original	7.29	7.29	5.47
20%	Basis Sharing	60	15.3	7.77
	CGC(Ours)	**54.53**	**14.9**	**7.57**
30%	Basis Sharing	97.4	23.86	9.69
	CGC(Ours)	**88.33**	**23.17**	**9.52**
40%	Basis Sharing	195.95	43.89	13.62
	CGC(Ours)	**175.55**	**41.49**	**13.48**
50%	Basis Sharing	509.3	98.92	21.3
	CGC(Ours)	**371.75**	**88.27**	**20.16**

Table 3. Scalability results of the PPL(↓) for larger-scale LLMs on WikiText-2.

Model	LLaMA-7B	LLaMA-13B	LLaMA-30B
SVD	20061	946.31	54.11
FWSVD	1630	OOM	OOM
SVD-LLM	7.94	6.61	5.63
Basis Sharing	7.74	6.51	5.47
CGC(Ours)	**7.07**	**6.21**	**5.33**

SVD-based baselines on both language modeling and zero-shot reasoning. Even under high compression, the model maintains stable per perplexity and accuracy, indicating that the proposed major-axis alignment and perturbation control effectively preserve key representational structures. On LLaMA-7B, our approach achieves lower perplexity and higher average zero-shot scores across all datasets, showing clear advantages in both linguistic coherence and reasoning generalization. The gap further widens at higher compression, confirming that aligning shared subspaces with low-curvature Hessian directions improves robustness.

Similar trends appear on LLaMA2-7B in Tabel 2, where our method achieves lower PPL than Basis-Sharing across all tested ratios. This suggests that the proposed geometric basis selection and adaptive amplitude modulation not only minimize distortion from rank reduction but also preserve global semantic consistency. These results demonstrate that CGC scales effectively to large models, preserving fluency and reasoning ability under strong compression.

Larger-scale LLMs. To verify adaptability on larger models, we extend experiments to LLaMA-7B, 13B, and 30B. As shown in Table 3, existing methods (FWSVD) fail to scale due to high memory cost(OOM), while ours achieves the lowest perplexity across all sizes.

Extreme compression. At extreme compression ratios of 70–80% for LLaMA-7B model, where performance degradation is inevitable for any method, our approach demonstrates significant relative gains on identical setting. Our method consistently outperforms Basis Sharing with lower perplexity on both C4 and WikiText-2 datasets (Table 4). Even at 80% compression, it reduces perplexity by over 50% on C4, showcasing the representational stability.

Table 4. Perplexity comparison for LLaMA-7B between our method and Basis Sharing under extreme compression ratios on C4 and WikiText-2.

Dataset	Basis Sharing	Ours	Ratio
C4	651.8314	**603.4069**	70%
WikiText-2	136.8194	**125.0952**	
C4	2465.999	**995.33**	80%
WikiText-2	624.0834	**424.8948**	

Table 5. Performance comparison of different compression methods across diverse LLM architectures.

Method	Qwen3-4B	OPT-6.7B	Mistral-7B
ASVD	-	82	10.21
SVD-LLMv2	-	13.46	-
Basis Sharing	15.98	11.79	7.57
CGC(Ours)	**15.02**	**11.68**	**7.49**

Diverse LLM Architectures. We evaluate CGC across a wide range of architectures to demonstrate its universality. First, as shown in Table 5, our method outperforms existing compression techniques (e.g., ASVD, Basis Sharing) on OPT-6.7B, Qwen3-4B, and Mistral-7B, achieving the lowest perplexity under a 20% compression ratio.

Furthermore, to verify robustness, Table 6 extends the evaluation to more advanced architectures. The results indicate that CGC maintains performance close to the original model across diverse families (Qwen, LLaMA3) and scales effectively from 0.6B to 8B parameters, showing strong generalization capabilities without model-specific tuning.

Table 6. Generalization evaluation across different, more advanced architectures and datasets.

Models	Qwen3 0.6B	Llama3.2 3B	Llama-3.1 8B
WikiText-2			
Orginal	21.19	7.82	7.23
CGC(Ours)	23.45	9.11	8.5
C4			
Orginal	30.19	11.3	11.62
CGC(Ours)	36.34	18.31	14.02

4.2.2. Vision Transformer Model

Table 7 reports results on ImageNet using Swin-Transformer and DeiT backbones at same compression ratio. Compared with SVD-based baselines such as GFM (Yu & Wu, 2023), FiPS, and LossFac (Zhang et al., 2024), our method attains the smallest accuracy drop at a comparable compression ra-

tio (28%). The post-sharing Top-1 remains nearly identical to the original model, showing that the proposed alignment mechanism effectively preserves representational capacity under strong compression. These results confirm that CGC generalizes well beyond language models, adapting effectively to vision backbones with hierarchical structures.

Table 7. Performance comparison of our method and existing sharing methods for Vision-Transformer on ImageNet.

Model	Method	Top1	Top1-Share	Top1-Drop
Swin-L	AAFM	86.25	85.73	-0.52
	GFM	86.25	85.83	-0.42
	FiPS	86.24	86.21	-0.03
	LossFac	86.23	86.19	-0.04
	Ours	86.24	86.23	**-0.01**
DeiT-B	PELA	81.80	79.46	-2.34
	Ours	81.87	81.84	**-0.03**

Table 8. Comparison of transfer learning results between multiple visual models and sharing model at different compression rates. Drop indicates the magnitude of performance drop.

	Model	Acc / Share(%)	Drop	F1 / Share(%)	Drop	Ratio
CIFAR-10	Swin-L	97.70/97.41	-0.39	97.67/97.15	-0.52	20%
	Swin-B	90.81/91.00	+0.20	90.77/91.44	+0.67	20%
	DeiT-B	92.90/91.90	-1.00	92.87/91.90	-0.97	30%
	Swin-B	90.81/91.36	+0.45	90.77/91.35	+0.58	30%
CIFAR-100	Swin-L	82.82/81.72	-1.10	81.52/80.66	-0.86	20%
	Swin-B	67.39/68.05	+0.66	65.35/66.06	+0.71	20%
	DeiT-B	72.80/70.30	-2.50	71.33/69.56	-1.78	30%
	Swin-B	67.39/67.47	+0.08	65.35/65.14	+0.21	30%

Transferring Ability. In Table 8, we transfer the shared model to 2 downstream tasks, including CIFAR-10/100. Consistent with the results on ImageNet, our method achieves accuracy on par with the original model (Dosovitskiy, 2020; Touvron et al., 2021) on these downstream tasks. This indicates that parameter sharing preserves the model's generalization capability.

4.3. Parameter Sharing Results Analysis.

Visualizing Symmetry-Induced Structure. In Figure 3(a), our method successfully represents a 32-layer network using only 12 shared bases, significantly fewer than the 16 used by Basis Sharing (BS) and the 32 in the non-sharing baseline. Figure 3(b) reveals the specific mapping on LLaMA-7B. Crucially, our method discovers deep, non-local redundancies that heuristic methods miss. For instance, layers 15, 17, and 18 are assigned the same color (basis) despite being non-adjacent. This proves that CGC breaks the limitation of "neighbor-only" sharing, capturing global structural similarities across the network.

Mathematically, layers with the same color form an orbit

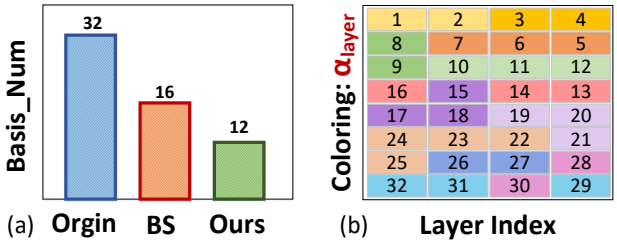

Figure 3. (a) Comparison of the number of basis in our method (32 layers represented by only 12 basis) with (Wang et al., 2025a). (b) Specific coloring scheme α_{layer} of our method when compressing LLaMA 7B by 50% (same color indicates shared basis).

under the automorphism group action. This means layers within a group (e.g., $\mathcal{L}_b$) are structurally interchangeable without affecting functional consistency. We quantify this symmetry using the group size $|\text{Aut}(\mathcal{G})| = \prod |\mathcal{L}_b|!$. As detailed in Appendix D, we observe a strong correlation: larger automorphism groups (higher symmetry) generally correspond to lower reconstruction error, validating that our method finds the most "natural" compression structure.

Validating Curvature-aware Effectiveness. To verify whether the performance gain stems from our curvature-based guidance or merely the basis generation process, Figure 4 validates the geometric effectiveness of our curvature-aware coloring. We compare the L2 reconstruction error of layers 7–10 across different parameter modules (Gate, K, Q, Up, V) under two strategies: heuristic adjacent grouping and our CGC, with the basis generation algorithm held constant. CGC yields consistently lower errors across all modules. This demonstrates that by aligning sharing perturbations with the Hessian's major axes, our method minimizes post-sharing distortion. This result confirms that the improvement is not solely due to the basis generation, but is directly attributable to the curvature-aware selection strategy, which identifies high-affinity layers that minimize distortion more effectively than heuristic proximity.

4.4. Computational Performance Experiments

Inference Efficiency on Real Hardware. Table 9 summarizes the inference efficiency of our method across three representative models, evaluated on a single NVIDIA A800 GPU with batch size 512 and sequence length 32. Across all models, our approach consistently reduces both parameter count and MACs by around 40–50%, leading to nearly 45% lower latency and up to 70% higher throughput compared with the original model. These results indicate that our method not only maintains model accuracy but also brings substantial runtime benefits, showing strong generalization across different architectures and demonstrating its practicality for large-scale deployment.

Hessian and Algorithm Time. We conducted a detailed

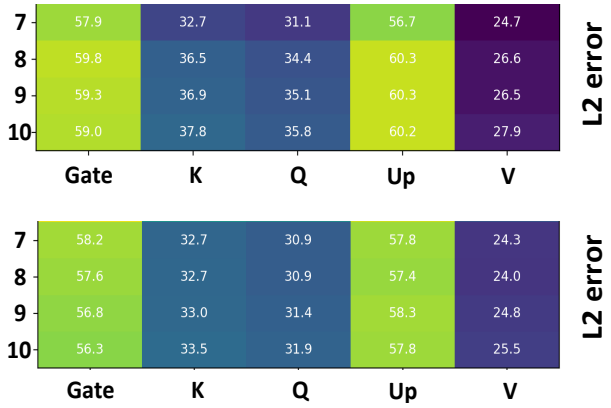

Figure 4. Comparison of L2 error caused by different coloring functions (top: adjacent, bottom: CGC), including the difference between the weights after sharing in layers 7-10 and the weights of the original standard model.

Table 9. Inference efficiency of our method on real hardware.

Model	Params.(B)	MACs(B)	Latency(s)	Throughput(t/s)
LLaMA2-7B	6.74B	6.61B	13.21	1338.37
Ours	**3.50** ↓48.1%	**3.94** ↓40.4%	**7.06** ↓46.6%	**2152.92** ↑60.9%
LLaMA-7B	6.74B	6.61B	13.27	1331.88
Ours	**3.99** ↓40.8%	**3.94** ↓40.4%	**7.38** ↓44.4%	**2084.30** ↑56.5%
Mistral-7B	7.24B	7.11B	14.61	1248.48
Ours	**3.75** ↓48.2%	**3.99** ↓43.9%	**7.93** ↓45.7%	**2135.37** ↑71.0%

breakdown of the computational overhead introduced by Hessian estimation, computed layer-wise for each linear module using a standard calibration subset. For a typical setup (batch size 1, context length 512, 20 iterations), the process involves two main stages:

Hessian Eigenvalue Computation: Calculating the top-550 eigenvalues takes approximately 0.93 hours.

Energy Minimization: The subsequent high-curvature energy minimization requires 0.4 hours.

This runtime is consistent with other SOTA Hessian-based methods. As a one-time offline cost, this "curvature budget" is highly efficient, trading marginal preparation time for superior fidelity without increasing inference latency.

Dynamic Tying (Hay & Wolf, 2024) take around **13.8 hours with PPL:49.37** on WikiText. CGC demonstrates significantly faster efficiency while achieving lower PPL (**42.30**). During deployment, our method maintains lower PPL while keeping inference time comparable to Basis Sharing.

4.5. Ablations

The Number of Short-Axis. Figure 5(a) examines the effect of the short-axis count t in Algorithm. Increasing t refines the estimation of high-curvature directions, allowing more accurate projection of perturbations onto the flat subspace. Consequently, perplexity decreases steadily, but computation time grows nearly linearly due to higher projection cost. This confirms that larger t improves the curvature fidelity of the alignment but at the expense of efficiency.

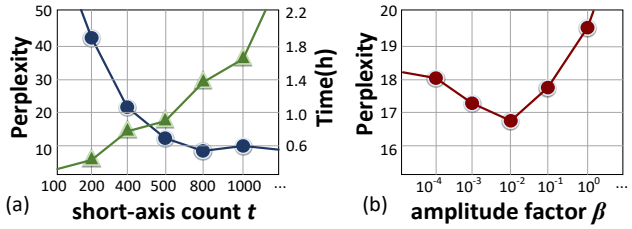

Figure 5. Ablations. (a) As the number of minor axes increases, perplexity consistently decreases, though computational burden increases. (b) When the amplitude factor increases, excessive perturbation leads to a sharp surge in perplexity.

The Amplitude Factor. Figure 5(b) investigates the amplitude factor β, which controls the trust-region radius in alignment. Small β overly restrict perturbations, preventing sufficient movement along flat directions and leading to underfitting. As β increases to around $10^{-2} - 10^{-1}$, moderate perturbation energy improves alignment and yields the lowest perplexity. Beyond this range, excessive amplitude breaks the local quadratic assumption and injects noise into sensitive directions, sharply degrading performance.

The First-Order Term. To verify the validity of the second-order approximation in Eq.1, we evaluate the first-order contribution in the Taylor expansion on ViT. We compute the ratio $c = 2\left|\nabla_W \mathcal{J}(W)^\top \delta\right| / \left|\delta^\top H \delta\right|$, for each layer and each sampled perturbation δ obtained from the sharing process. Empirically, $c < 0.04$ ($c < 0.1$ in LLaMA) for 90% of layers, indicating that the first-order term is negligible compared with the second-order curvature term. This phenomenon arises because the model is already well optimized—the gradient norm $\|\nabla_W \mathcal{J}\|$ is close to zero—making the first-order term vanish at convergence. Consequently, the loss change is dominated by the second-order, which validates the assumption used in Eq.1 and supports our sharing strategy.

5. Conclusion

This work presents Curvature-aware Adaptive Grouping (CGC), a structured framework that reformulates parameter sharing by aligning layer subspaces with low-curvature Hessian directions. Grounded in symmetry insights, CGC replaces heuristic rules with a principled, training-free strategy that offers a unified view of cross-layer redundancy. Experiments on vision and language models confirm that CGC achieves superior compression–accuracy trade-offs compared to both heuristic and SVD-based baselines.

6. Impact Statements

The goal of this paper is to advance the field of machine learning compression. However, our work has no potential negative social impact and is only intended to provide a reference for future research in this area.

References

Amini, A., Gabriel, S., Lin, P., Koncel-Kedziorski, R., Choi, Y., and Hajishirzi, H. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*, 2019.

Banerjee, P., Pal, K. K., Mitra, A., and Baral, C. Careful selection of knowledge to solve open book question answering. *arXiv preprint arXiv:1907.10738*, 2019.

Bisk, Y., Zellers, R., Gao, J., Choi, Y., et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Dehghani, M., Gouws, S., Vinyals, O., Uszkoreit, J., and Kaiser, Ł. Universal transformers. *arXiv preprint arXiv:1807.03819*, 2018.

Dosovitskiy, A. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Frantar, E., Singh, S. P., and Alistarh, D. Optimal brain compression: A framework for accurate post-training quantization and pruning, 2023. URL https://arxiv.org/abs/2208.11580.

Golub, G. H., Hoffman, A., and Stewart, G. W. A generalization of the eckart-young-mirsky matrix approximation theorem. *Linear Algebra and its applications*, 88:317–327, 1987.

Hassibi, B., Stork, D., and Wolff, G. Optimal brain surgeon and general network pruning. In *IEEE International Conference on Neural Networks*, pp. 293–299 vol.1, 1993. doi: 10.1109/ICNN.1993.298572.

Hay, T. D. and Wolf, L. Dynamic layer tying for parameter-efficient transformers. *arXiv preprint arXiv:2401.12819*, 2024.

Hsu, Y.-C., Hua, T., Chang, S., Lou, Q., Shen, Y., and Jin, H. Language model compression with weighted low-rank factorization, 2022. URL https://arxiv.org/abs/2207.00112.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Lv, X., Zhang, P., Li, S., Gan, G., and Sun, Y. Lightformer: Light-weight transformer using svd-based weight transfer and parameter sharing. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 10323–10335, 2023.

Marcinkiewicz, M. A. Building a large annotated corpus of english: The penn treebank. *Using Large Corpora*, 273: 31, 1994.

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

Qwen, :, Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Tang, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., and Qiu, Z. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

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

Reid, M., Marrese-Taylor, E., and Matsuo, Y. Subformer: Exploring weight sharing for parameter efficiency in generative transformers. *arXiv preprint arXiv:2101.00234*, 2021.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention, 2021. URL https://arxiv.org/abs/2012.12877.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, J., Chen, Y., Lin, I., Li, B., and Zhang, G. Basis sharing: Cross-layer parameter sharing for large language model compression. In *13th International Conference on Learning Representations, ICLR 2025*, 13th International Conference on Learning Representations, ICLR 2025, pp. 39540–39558. International Conference on Learning Representations, ICLR, 2025a. Publisher Copyright: © 2025 13th International Conference on Learning Representations, ICLR 2025. All rights reserved.; 13th International Conference on Learning Representations, ICLR 2025 ; Conference date: 24-04-2025 Through 28-04-2025.

Wang, Q., Ke, J., Tomizuka, M., Chen, Y., Keutzer, K., and Xu, C. Dobi-svd: Differentiable svd for llm compression and some new perspectives. *arXiv preprint arXiv:2502.02723*, 2025b.

Wang, X., Zheng, Y., Wan, Z., and Zhang, M. Svd-llm: Truncation-aware singular value decomposition for large language model compression, 2025c. URL https://arxiv.org/abs/2403.07378.

Wu, Y., Kan, S., Zeng, M., and Li, M. Singularformer: Learning to decompose self-attention to linearize the complexity of transformer. In *IJCAI*, pp. 4433–4441, 2023.

Yu, H. and Wu, J. Compressing transformers: features are low-rank, but weights are not! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 11007–11015, 2023.

Yuan, Z., Shang, Y., Song, Y., Yang, D., Wu, Q., Yan, Y., and Sun, G. Asvd: Activation-aware singular value decomposition for compressing large language models, 2025. URL https://arxiv.org/abs/2312.05821.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Zhang, B., Cheng, D., Zhang, Y., Liu, F., and Tian, J. Lossless model compression via joint low-rank factorization optimization. *arXiv preprint arXiv:2412.06867*, 2024.

Zhang, B., Cheng, D., Zhang, Y., Liu, F., and Chen, W. Compression for better: A general and stable lossless compression framework, 2025. URL https://arxiv.org/abs/2412.06868.

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

Üyük, C., Lasby, M., Yassin, M., Evci, U., and Ioannou, Y. Learning parameter sharing with tensor decompositions and sparsity, 2025. URL https://arxiv.org/abs/2411.09816.