LaMoC: Loss-Aware Modular Compression for LLMs

Mohanad Odema[*]
LG Electronics USA

Jacob Song
LG Electronics USA

Abstract

Modular compression has enabled considerable parameter reduction in LLMs while preserving strong language understanding and downstream task accuracy. However, existing joint modular compression methods primarily rely on activation statistics, leaving loss-sensitivity information and its module-level characterization underexplored. We investigate addressing this gap with LaMoC, a loss-aware modular compression methodology that blends activation and Empirical Fisher statistics through gradient-error alignment. LaMoC improves joint compression by selecting compression statistics that better align local module reconstruction error with the downstream loss. Our contributions are three-fold: (1) We characterize the Empirical Fisher as a module-level loss-aware proxy that can be blended with the activation statistics required for compression. (2) We reformulate joint modular compression as a two-tiered optimization problem that minimizes module reconstruction error while tuning the activation and gradient information blending rate. (3) We implement an empirically driven methodology with statistical validation to solve the resulting compression problem. We evaluate LaMoC across four model families spanning eight models. On the 4–8B models, LaMoC achieves an average 2.5% reduction in perplexity and a 1% relative improvement in task accuracy over state-of-the-art modular compression methods.

1 Introduction

Parameter reduction is critical to enabling LLM efficiency in constrained deployment environments and at scale. Methods like pruning (Ma et al., 2023; Ashkboos et al., 2024; Frantar and Alistarh, 2023; Bai et al., 2024) and low-rank approximation (Wang et al., 2025c; Yuan et al., 2023; Wang et al., 2025a) have gained widespread adoption. Pruning removes weights, heads, or hidden dimensions based on a metric of importance in a structured or an unstructured manner, whereas low-rank approximation replaces the dense weight matrices with two low-rank factors that are more parameter- and compute-efficient. For example, Singular Value Decomposition (SVD) can approximate a single weight matrix as $W_{m\times n} \approx A_{m\times r}B_{r\times n}$ such that $r \ll \min(m, n)$, effectively reducing parameter count while providing optimal rank-r truncation under Eckart-Young-Mirsky theorem (Eckart and Young, 1936; Mirsky, 1960).

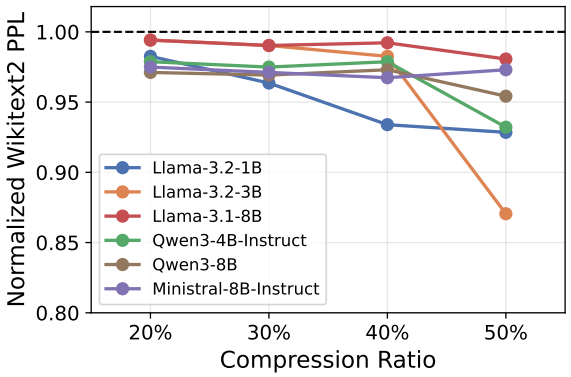


Figure 1: Normalized WikiText-2 perplexity for LaMoC relative to baseline MoDeGPT (Lin et al., 2025).

Until recently, the considerable loss in knowledge capacity and task proficiency as a result of parameter reduction often necessitated a subsequent fine-tuning stage post compression. However, recent advances (Lin et al., 2025; Liu et al., 2025; Yin et al., 2025; Chiang et al., 2026) have substantially closed the post compression performance gap, reducing the reliance on expensive fine-tuning procedures, with several methods positioned as training-free approaches with an optional fine-tuning recovery stage. The effectiveness of these training-free methodologies can be attributed to these ideas:

- **Activation Awareness.** Modern compression methods increasingly move beyond weight-only

[*]Corresponding author: mohanad.odema@lge.com

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 28280–28294
October 24-29, 2026 ©2026 Association for Computational Linguistics

criteria by using activation statistics to guide compression. This information can define the reconstruction target in low-rank approximation (Wang et al., 2025c; Yuan et al., 2023) or rank parameter importance for pruning (Sun et al., 2024).

- **Loss Sensitivity Information.** Another thread follows the ideas of Optimal Brain Damage (LeCun et al., 1989; Hassibi et al., 1993), where loss-derivative statistics are incorporated to assess parameter importance for compression, applicable in both pruning (Ma et al., 2023; Yin et al., 2025) and low-rank reconstruction through loss sensitivity estimates (Hsu et al., 2022).
- **Non-uniform Truncation.** Different layers and weight groups exhibit varying degrees of sensitivity to compression. Thus, recent approaches have opted to assign pruning/truncation ratios non-uniformly based on a measure of such sensitivity (Men et al., 2025; Yang et al., 2025b; Li et al., 2025; Wang et al., 2025b; Lin et al., 2025).
- **Modularity.** Rather than compressing each weight matrix independently, modular compression jointly compresses related weight groups, enabling module-level reconstruction objective and tailoring compression operators to each module's structural and functional properties (Lin et al., 2025; Wong et al., 2025; Chiang et al., 2026; Koike-Akino et al., 2025).

The combination of all four techniques remains underexplored – specifically the incorporation of loss sensitivity information for modular compression. For instance, the state-of-the-art training-free compression method, MoDeGPT (Lin et al., 2025), is a modular compression approach that leverages activation awareness and non-uniform truncation. In Figure 1, we motivate how further consideration of loss sensitivity information on the module level can better preserve language modeling capabilities post compression, as seen by the decrease in perplexity compared to baseline MoDeGPT across different model architectures. From here, we frame the following research questions:

- ***RQ1:*** *Can loss-sensitivity information be systematically incorporated into modular compression to improve language modeling and downstream task performance?*
- ***RQ2:*** *How can module-level loss sensitivity be characterized and blended with the activation statistics required for joint compression?*
- ***RQ3****: How can joint modular compression be formulated as a loss-aware optimization problem to be solved in practical settings?*

Given these RQs, we summarize the key contributions of this work as follows:

- We derive a module-level characterization to incorporate loss sensitivity information into modular compression using gradient-weighted second order activation statistics.
- We formulate the loss-aware modular compression objective as a two-tiered optimization problem minimizing the modular reconstruction loss while selecting a per-module blending rate between activation and gradient-weighted statistics.
- We present LaMoC, an empirically-driven methodology with statistical validation to address the resulting two-tiered loss-aware modular compression problem leveraging fixed and adaptive strategies for controlling the degree of gradient information blending in a practical manner.
- We conduct experiments across 4 model families and 8 models showing that LaMoC improves SOTA modular compression pipelines by on average 2.5% and 1% in language modeling (perplexity) and downstream task accuracy, respectively.

2 Background

2.1 Related Works

Parameter Reduction. Parameter reduction aims to improve LLM efficiency and memory footprint by removing or approximating less important parameters. This is commonly achieved through pruning (Molchanov et al., 2019; Xia et al., 2022; Ashkboos et al., 2024; Frantar and Alistarh, 2023; Sengupta et al., 2025; Ma et al., 2023; Men et al., 2025) and low-rank approximation (Wang et al., 2025c,b,a; Yuan et al., 2023; Hu et al., 2026; Hsu et al., 2022). Typically, pruning removes structured components such as layers, modules, channels, heads, or hidden dimensions, while low-rank approximation replaces dense weight matrices with compact factors. These methods define the basic parameter reduction operators. Recently, the application of these compression operators has extended beyond isolated matrices to coupled weight groups.

Modular Compression. Recent works (Lin et al., 2025; Chiang et al., 2026; Wong et al., 2025; Koike-Akino et al., 2025) advance this direction through modular compression, where pruning or

low-rank approximation is applied to structured groups of weight matrices according to their functional role and architectural properties. This modular view better reflects the underlying dynamics of coupled matrices whose outputs interact through non-linear operations and downstream transformations. As a result, modular compression introduces a broader module-level reconstruction objective rather than treating each matrix independently. Still, modular methods rely predominantly on activation-driven reconstruction criteria to guide the compression objective.

Activation-Aware Compression. Instead of minimizing weight reconstruction error, modern parameter-reduction methods (Wang et al., 2025c,b,a; Hu et al., 2026; Lin et al., 2025; Chiang et al., 2026) target the reconstruction of output feature representations: $\min \|X_l(W_l - \hat{W}_l)\|_F$, where X_l is the input to layer l, and W_l and $\hat{W}_l$ denote the original and compressed weights. Typically to achieve this in practice, calibration data is propagated through the model to collect second-order activation statistics (Gram autocorrelations), which are then used to project weights into the activation space or identify important dimensions. Recent works (Odema et al., 2026) study the role of calibration data in residual error accumulation during compression and its effect on layer-sensitivity misalignment. Still, these activation statistics do not necessarily capture how compression decisions affect the downstream loss.

Loss-Aware Compression. A complementary line of work incorporates loss-derivative information to guide the compression decisions. Early methods such as Optimal Brain Damage and Optimal Brain Surgeon (LeCun et al., 1989; Hassibi et al., 1993) use second-order derivative information to guide pruning decisions. The idea still persists in recent pruning (Singh and Alistarh, 2020; Ma et al., 2023; Frantar and Alistarh, 2023) and low-rank approximation (Hsu et al., 2022) methods, where Hessian- or Fisher-based curvature proxies are used to estimate compression sensitivity and weight parameter importance.

Motivation. Despite progress in modular and loss-aware compression, their combination remains underexplored. Existing modular methods primarily rely on activation-driven reconstruction objectives, while loss-aware methods are often applied at the level of individual parameters or matrices. This motivates a study on loss-aware modular compression, how to incorporate loss-curvature statistics into module-wise compression objectives, and the potential gains from this approach. More details on the positioning of this work in relation to other loss-aware frameworks are in Appendix B.

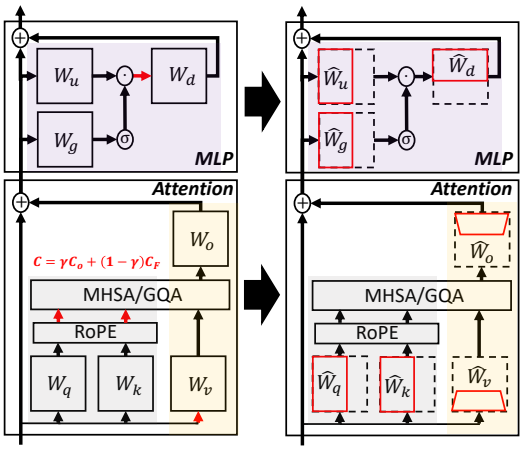


Figure 2: LLM Modular Compression. (*left*) Colors demonstrate the different module groupings (QK, VO, MLP). Red arrows indicate the activation space in which the activation Gram $\mathcal{C}_0$ is collected and the empirical Fisher-weighted Gram $\mathcal{C}_F$ is projected into that same space from gradients computed at each module's output projection. The combined gradient-weighted Gram formulation is also shown in red. (*right*) Weight matrices post Modular Compression.

2.2 Preliminaries on Joint Decomposition

Figure 2 illustrates the key module constructions for joint compression following recent SOTA works (Lin et al., 2025; Chiang et al., 2026; Wong et al., 2025) detailed in the following.

Notation. Let d_h denote the hidden dimension of the transformer model; $d_{\text{head}} = \frac{d_h}{n_{\text{head}}}$ denote the head dimension given n_{head} attention heads[1]; d_{int} denote the intermediate dimension. The up and gate projection weight matrices are denoted by $W_u, W_g \in \mathbb{R}^{d_h \times d_{\text{int}}}$, respectively; the down projection weight matrix is denoted by $W_d \in \mathbb{R}^{d_{\text{int}} \times d_h}$; The query, key, and value matrices are denoted by $W_q, W_k, W_v \in \mathbb{R}^{d_h \times (n_{\text{head}} \times d_{\text{head}})}$; The output projection matrix $W_o \in \mathbb{R}^{(n_{\text{head}} \times d_{\text{head}}) \times d_h}$. Broadly, activation Gram, matrices are denoted by $\mathcal{C} = X^\top X \in \mathbb{R}^{d \times d}$ where $X \in \mathbb{R}^{T \times d}$ indicate the input sequence of length T and a dimension d specified based on the where X is captured. $\rho(\cdot)$ indicates Rotary Positional Embedding (RoPE).

MLP Module. W_u, W_g, and W_d are jointly compressed through *dimension pruning* of the shared intermediate dimension d_{int}, reducing it to

[1]For simplicity, Notation shown is for multi-head attention

$r \ll d_{\text{int}}$. In order to select which intermediate dimensions to prune, the following sequence is used: 1) Activation gram statistics $\mathcal{C}_{\text{int}} = X_{\text{int}}^\top X_{\text{int}} \in \mathbb{R}^{d_{\text{int}} \times d_{\text{int}}}$ are collected prior to down projection using calibration data; 2) Per-dimension scores $\mathbf{s} \in \mathbb{R}^{d_{\text{int}}}$ are computed based on criterion like ridge leverage scores $\mathbf{s} = \text{diag}(\mathcal{C}_{\text{int}}(\mathcal{C}_{\text{int}} + \lambda I)^{-1}) \in \mathbb{R}^{d_{\text{int}}}$; 3) The intermediate dimensions are sorted in a descending importance order based on $\mathbf{s}$; 4) A selection matrix from top-r scores $S_k \in \mathbb{R}^{d_{\text{int}} \times r}$ is constructed to be multiplied by the individual weight matrices to prune d_{int} providing $W_u S_k$, $W_g S_k \in \mathbb{R}^{d_h \times r}$ and $S_k^\top W_d \in \mathbb{R}^{r \times d_h}$.

QK Module. W_q and W_k are truncated along the head dimension d_{head}. The key idea is to derive per-head scoring vectors, $\mathbf{s}_{\text{head}} \in \mathbb{R}^{d_{\text{head}}}$, before applying *dimension pruning*. The scores are estimated based on channel importance evaluated at each attention head. Specifically, given an attention block input, $X \in \mathbb{R}^{T \times d_h}$, activation channel correlations are collected at each head i through:

- **Query Correlations.** Let $X_q^i = \rho(XW_q^i)$ be the RoPE-transformed query activation for the i^{th} attention head. Then, the query autocorrelation is $\mathcal{C}_q^i = (X_q^i)^\top X_q^i \in \mathbb{R}^{d_{\text{head}} \times d_{\text{head}}}$.
- **Key Correlations.** Let $X_k^i = \rho(XW_k^i)$ be the RoPE-transformed key activation for the i^{th} attention head. Then, the key autocorrelation is $\mathcal{C}_k^i = (X_k^i)^\top X_k^i \in \mathbb{R}^{d_{\text{head}} \times d_{\text{head}}}$.
- **QK Correlations.** Given $\mathcal{C}_k^i$ and $\mathcal{C}_q^i$, the final importance scores for each head i are evaluated through a Hadamard product of the column norm vectors of their matrix square roots, $\mathbf{s}^i = ||(\mathcal{C}_q^i)^{1/2}||_{\text{col}} \odot ||(\mathcal{C}_k^i)^{1/2}||_{\text{col}} \in \mathbb{R}^{d_{\text{head}}}$.

Using $\mathbf{s}^i$, a per-head selection matrix $S_{qk}^i \in \mathbb{R}^{d_{\text{head}} \times r}$ with $r \ll d_{\text{head}}$ can be constructed to select the top-r important channels: $\hat{W}_q^i = W_q^i S_{qk}^i$ and $\hat{W}_k^i = W_k^i S_{qk}^i$, where $\hat{W}_q^i, \hat{W}_k^i \in \mathbb{R}^{d_h \times r}$.

VO Module. W_v and W_o are jointly compressed through a sequence of SVD operations which render two low-rank matrices $\hat{W}_v \in \mathbb{R}^{d_h \times (n_{\text{head}} \times r)}$ and $\hat{W}_o \in \mathbb{R}^{(n_{\text{head}} \times r) \times d_h}$. To achieve this, activation Gram statistics are collected at the input to W_v, yielding $\mathcal{C}_v = X_v^\top X_v \in \mathbb{R}^{d_h \times d_h}$. For each head, two successive SVD operations are applied to approximate $\mathcal{C}_v^{1/2} W_v^i W_o^i$ as follows: (1) $(\mathcal{C}_v^{1/2} W_v^i) W_o^i = (U_v^i \Sigma_v^i (V_v^i)^\top) W_o^i$; (2) $U_v^i (\Sigma_v^i (V_v^i)^\top W_o^i) \approx U_v^i (U_r^i \Sigma_r^i (V_r^i)^\top)$, where the subscript r denotes truncation to a rank r. The final low-rank matrices can then be defined as follows: $\hat{W}_v^i \leftarrow \mathcal{C}_v^{-1/2} U_v^i U_r^i$ and $\hat{W}_o^i \leftarrow \Sigma_r^i (V_r^i)^\top$.

3 LaMoC: Loss-Aware Modular Compression for LLMs

This section introduces our method for loss-aware modular compression, which enforces gradient-error alignment during compression to incorporate loss sensitivity into the reconstruction objective.

Key Contribution and Overview. The key contribution here lies in the formulation of loss-aware low-rank compression in the module activation space by blending gradient-weighted and canonical activation statistics, where a two-tiered compression objective is derived spanning module reconstruction and the gradient information blending rate. Based on empirical observations, we then propose fixed and adaptive blending strategies to control the blending rate in the module activation space according to an estimate on the expected change in cross entropy.

3.1 Gradient-Error Alignment

Following the characterization of gram matrices for joint compression in Section 2.2, we define a gradient-weighted effective Gram matrix. For readability, we omit the module superscript:

Definition 1. *Let $\mathcal{C}_0 = (X)^\top X$ be a canonical activation Gram matrix collected for an arbitrary module l. Define $\mathcal{C}_F$ as a Fisher-weighted activation gram, then an effective gradient-aligned activation Gram matrix is given as:*

$$\mathcal{C}_{\text{eff}}(\gamma) = \gamma \mathcal{C}_0 + (1 - \gamma)\mathcal{C}_F \tag{1}$$

where $\gamma \in [0, 1]$ is a blending coefficient to control the degree of gradient information blending.

The red arrows in Figure 2 indicate for each module the activation space in which $\mathcal{C}_0$ and $\mathcal{C}_F$ are blended. This gradient-aligned Gram formulation can replace any activation Gram derived for any module from Section 2.2. The canonical activation gram without any gradient information blending is obtained at $\gamma = 1$. For $\gamma < 1$, $\mathcal{C}_F$ skews the effective autocorrelation towards channels or energy directions more sensitive to the downstream loss. To provide a formal definition for $\mathcal{C}_F$, we first define a diagonal Fisher approximation as a per-channel importance vector:

Definition 2. *Let $\delta_n = \frac{\partial \mathcal{L}_{\text{CE}}}{\partial x_n} \in \mathbb{R}^d$ be the gradient of cross entropy with respect to activation vector x_n*

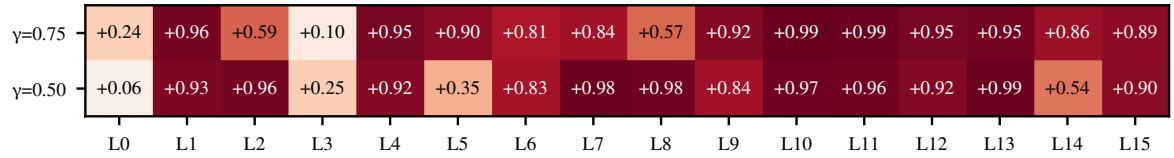


Figure 3: Pearson Correlation (↑) between fixed reference first-order proxy and true $\Delta\mathcal{L}_{\text{CE}}$ for Llama-3.2-1B layers.

for the n^{th} calibration sample at a target compression module. Then, per-channel importance can be derived using the diagonal of empirical Fisher:

$$\mathbf{f} = \text{diag}(\mathbb{E}_n[\delta_n(\delta_n)^\top]), \qquad \mathbf{f} \in \mathbb{R}^d \tag{2}$$

which ignores cross-channel gradient correlations following the approximation in (LeCun et al., 1989; Hassibi et al., 1993).

From the diagonal Fisher approximation we can construct a Fisher-weighted activation Gram:

Definition 3. *Let $\mathbf{f} \in \mathbb{R}^d$ be the diagonal empirical Fisher vector characterizing the per-channel importance for a given module, then the Fisher-weighted activation Gram is defined as:*

$$\mathcal{C}_F = D_f^{1/2}\mathcal{C}_0 D_f^{1/2}, \qquad D_f^{1/2} = \text{diag}(\sqrt{\mathbf{f}}) \tag{3}$$

Lastly, we rescale $\mathcal{C}_F$ using a trace ratio to match the total energy of canonical $\mathcal{C}_0$ before blending:

$$\mathcal{C}_F \leftarrow \mathcal{C}_F \frac{\text{trace}(\mathcal{C}_0)}{\text{trace}(\mathcal{C}_F)} \tag{4}$$

This ensures γ blending between the canonical and Fisher-weighted statistics in Equation 1 is conducted on a matched energy scale. This concludes the gradient-aware effective Gram construction.

3.2 Problem Formulation

Given the loss-sensitive effective Gram matrix construction in Section 3.1, the joint compression output reconstruction objective can be redefined.

Definition 4. *Let m be a target module to be jointly compressed, and let $\mathcal{C}_{\text{eff}}(\gamma_m)$ denote the effective Gram defined in Equation 1. The module output reconstruction objective in a loss-aware joint compression process can be defined through the following two-tiered optimization formulation:*

$$\hat{m}^\star = \arg\min_{\hat{m}} ||(\hat{m} - m)\mathcal{C}_{\text{eff}}^{1/2}(\gamma_m^\star)||_F^2 \tag{5}$$

$$\gamma_m^\star = \arg\min_{\gamma_m \in [0,1]} \Delta\mathcal{L}_{\text{CE}}(\gamma_m) \tag{6}$$

where $\hat{m}^\star$ represents the final compressed module conditioned on an optimal choice of $\gamma_m^\star$ that minimizes degradation in cross-entropy loss $\Delta\mathcal{L}_{\text{CE}}(\gamma_m)$ relative to canonical compression.

3.3 Method

Solving the two-tiered optimization objective requires addressing the following challenges:

- Navigating a combinatorial solution space comprising layers, modules, and γ options
- Evaluation of $\Delta\mathcal{L}_{\text{CE}}$ and the gradients for each candidate compression solution

We present our LaMoC method derived based on empirical analysis and statistical observations.

First-Order Loss Approximation. (LeCun et al., 1989) and (Hassibi et al., 1993) used a Taylor Expansion to formulate the change in loss due to parameter reduction, generally defined as:

$$\Delta\mathcal{L}_{\text{CE}} = \delta^\top \Delta w + \frac{1}{2}\Delta w^\top \mathcal{H} \Delta w \tag{7}$$

where δ indicates the gradient $\frac{\partial \mathcal{L}_{CE}}{\partial w}$ with respect to perturbed parameter; Δw denotes parameter perturbation; and $\mathcal{H}$ the Hessian matrix. In our formulation, $\Delta\mathcal{L}_{CE}$ is only required to identify $\gamma_m^\star$ for a module m as per Equation 6. Therefore, we can leverage a First-order approximation to evaluate the Expected $\Delta\mathcal{L}_{CE}$ post module m compression without materializing the Hessian as follows:

$$\mathbb{E}[\Delta\mathcal{L}_{\text{CE}}(\gamma_m)] = \frac{1}{N}\sum_{n=1}^{N}(\delta_n^m)^\top X_n(\hat{m}_{\gamma_m} - m) \tag{8}$$

where N is the number of input tokens; X_n is the n^{th} input at module m; δ_n^m is the gradient with respect to the n^{th} module output; $\hat{m}_{\gamma_m}$ is compressed module version given γ_m.

Reference Gradients. To avoid evaluating the δ_n^m with every sampled γ in Equation 8, we use the canonical solution gradients at γ=1 as the reference gradients to be used for all γ solutions at a module m. Thus, the expected change in cross-entropy in Equation 8 becomes relative to the canonical solution whose gradients are only computed once, and reused for subsequent candidates. This enables faster evaluation without expensive backpropagation requirements for each γ candidate. The reference gradients for proxy prediction are computed

Method	Llama-3.1-8B				Ministral-8B-It-2410				Qwen3-4B-It-2507				Qwen3 8B			
	20%	30%	40%	50%	20%	30%	40%	50%	20%	30%	40%	50%	20%	30%	40%	50%
UniQL (Chiang et al., 2026)	9.20	12.18	18.29	30.15	9.79	12.18	17.18	27.03	14.93	22.76	43.87	168.17	14.47	19.47	45.98	196.62
MoDeGPT (Lin et al., 2025)	8.61	10.79	15.19	22.89	9.05	10.75	14.33	20.60	12.28	15.77	25.79	90.02	12.18	15.13	28.55	83.58
Fixed $\gamma = 0.5$	**8.54**	**10.67**	**15.07**	22.54	**8.83**	10.46	13.89	**20.05**	12.06	15.52	25.54	92.15	11.88	14.75	28.94	84.89
Fixed $\gamma = 0.75$	8.55	10.71	**15.07**	**22.45**	8.86	10.50	13.94	20.16	12.13	15.67	**25.24**	89.32	12.06	14.90	28.27	82.60
Adaptive $\gamma^\star$	8.55	10.71	**15.07**	**22.45**	**8.83**	**10.44**	**13.86**	**20.05**	**12.02**	**15.37**	**25.24**	**83.91**	**11.83**	**14.67**	**27.78**	**79.75**

Table 1: WikiText-2 perplexity (↓) using 128 calibration samples from WikiText-2 at 2048 sequence length.

Method	Llama-3.2-3B		Llama-3.2-1B	
	20%	40%	20%	40%
UniQL (Chiang et al., 2026)	14.93	47.44	27.03	184.70
MoDeGPT (Lin et al., 2025)	10.88	22.32	17.52	44.21
Fixed $\gamma = 0.5$	10.84	22.02	16.98	40.97
Fixed $\gamma = 0.75$	10.86	21.97	17.05	41.70
Adaptive $\gamma^\star$	10.81	21.93	17.21	41.29

Table 2: WikiText-2 PPL (↓) for Llama-3.2-3B and Llama-3.2-1B models.

at the (γ=1.0) compression, and the activations are not updated to consider previous layer effects.

Statistical Validation. We validate the usage of fixed reference gradients in the first-order proxy for estimating $\Delta\mathcal{L}_{\text{CE}}$ using a statistical analysis measuring Pearson correlation between the proxy estimates and the true cross-entropy changes. Figure 3 illustrates the correlation at each layer's last module output for a Llama-3.2-1B model given $\gamma \in \{0.5, 0.75\}$. We observe the correlation strength differs per-layer as the proxy is dependent on the calibration settings and relies on a first-order approximation of reference gradients. Overall, we observe Pearson correlation reaches 0.91±0.28.

γ Selection Policy. Denote $\mathbf{p}(\gamma_m) = \mathbb{E}[\Delta\mathcal{L}_{\text{CE}}(\gamma_m)]$. Then, a heuristic γ selection policy per module can be defined:

$$\gamma_m^\star = \begin{cases} \arg\min_{\gamma_m \in \Gamma_m} \mathbf{p}(\gamma_m), & \text{if } \min_{\gamma_m \in \Gamma_m} \mathbf{p}(\gamma_m) < 0, \\ 1, & \text{otherwise.} \end{cases} \tag{9}$$

where Γ_m represents the set of viable discrete γ candidates under exploration. The condition ensures that γ is selected for the target module iff the proxy predicts a reduction in cross entropy loss relative to the canonical compression. Otherwise, canonical $\gamma_m = 1$ is assigned to the module m.

End-to-end Compression. Once $\gamma_m^\star$ is selected for each module, compression proceeds by applying the corresponding module-specific effective Gram and assembling the resulting modules into a gradient-aligned compressed model. We further validate the effects of compounded predictions on $\mathbf{p}(\gamma_m)$ for all modules of Llama-3.2-1B after 15% compression, and find that the overall Pearson correlation still holds at 0.913.

4 Experiments

Implementation. Unless otherwise stated, we build our loss-aware modular compression on top of MoDeGPT with groupings of MLP, QK, and VO following Section 2.2 supplemented by LaMoC implementation in Section 3.3. We follow MoDeGPT experimental setup with 128 calibration samples from WikiText-2 at sequence length 2048.

Selection Strategy. We fix $\gamma_l \in \{0.5, 0.75\} \forall l \in \{1, \cdots, L\}$ and adopt two strategies for γ selection: (1) *Fixed*; where the same γ value is assigned within each module in every layer. (2) *Adaptive*; where each module selects $\gamma^\star$ value from the candidate set which achieves the minimum expected cross entropy loss following Equation 9.

Baselines. Our key baselines are MoDeGPT (Lin et al., 2025) and UniQL (Chiang et al., 2026). Both works follow in principle the joint decomposition flow introduced in Section 2.2 with some differences as UniQL is more system-oriented and relaxed some of MoDeGPT principled methodology. We focus on MoDeGPT and UniQL as they represent training-free methods shown to have outperformed strong pruning baselines (SliceGPT (Ashkboos et al., 2024), LLM Surgeon(van der Ouderaa et al., 2024), ShortGPT(Men et al., 2025)).

Evaluation and Models. We evaluate the efficacy of our approach using WikiText-2 perplexity (Merity et al., 2016), 5-shot MMLU, and Zero-shot accuracy from the Lm-eval harness (Gao et al., 2023) – PiQA, Arc-E, Winogrande (accuracy); Hellaswag and Arc-C (normalized accuracy). We focus on the training-free compression performance following (Lin et al., 2025), and target 1B-8B parameter models across the Llama-3 (Grattafiori et al., 2024), Qwen3 (Yang et al., 2025a), and

Model	Compress.	Method	MMLU (5-shot)	ARC-e	ARC-c	PIQA	WinoG.	HellaS.	Average
Llama-3.1 -8B	0%	Dense	65.60	81.57	53.75	80.25	74.35	78.89	72.40
	20%	MoDeGPT	55.83	69.74	41.04	71.98	71.74	67.45	62.96
		Fixed $\gamma = 0.75$	56.20	71.17	41.04	72.03	71.03	66.60	63.01
		Adaptive $\gamma^\star$	57.04	70.96	40.78	71.87	70.56	66.62	62.97
	40%	MoDeGPT	41.61	48.53	29.78	64.31	63.54	48.89	49.44
		Fixed $\gamma = 0.75$	43.96	50.63	29.95	63.44	64.09	48.62	50.12
		Adaptive $\gamma^\star$	43.80	50.80	28.67	62.73	62.75	48.40	49.52
Ministral -8B-It -2410	0%	Dense	65.07	81.99	54.95	80.96	75.45	79.14	72.93
	20%	MoDeGPT	54.47	70.41	41.98	71.98	67.80	64.67	61.88
		Fixed $\gamma = 0.75$	55.88	70.66	41.98	71.49	68.82	64.62	62.24
		Adaptive $\gamma^\star$	55.95	70.79	42.24	71.55	68.90	64.72	62.36
	40%	MoDeGPT	31.56	49.28	29.61	62.13	62.51	45.14	46.71
		Fixed $\gamma = 0.75$	31.44	49.58	29.10	61.92	62.98	44.82	46.64
		Adaptive $\gamma^\star$	31.94	50.17	29.27	62.02	62.12	44.98	46.75
Qwen3-4B -It-2507	0%	Dense	72.60	83.21	58.70	76.01	67.88	69.09	71.25
	20%	MoDeGPT	46.26	70.92	46.33	72.09	62.75	62.77	60.19
		Fixed $\gamma = 0.75$	50.48	69.91	45.65	72.25	63.77	62.56	60.77
		Adaptive $\gamma^\star$	48.65	73.57	47.61	71.55	63.77	62.62	61.29
	40%	MoDeGPT	24.65	46.21	29.27	61.37	55.96	42.69	43.36
		Fixed $\gamma = 0.75$	24.90	48.27	30.55	62.46	56.20	42.89	44.21
		Adaptive $\gamma^\star$	24.80	48.86	29.35	62.46	55.88	42.87	44.04

Table 3: Task Accuracy at 20% and 40% training-free compression using 128 samples from WikiText-2.

Mistral (Mistral AI, 2024) families. We use base (Llama-3.1-8B, Llama-3.2-3B, Llama-3.2-1B) and instruct (Ministral-8B-Instruct-2410, Qwen3-4B-Instruct-2507[2], Qwen3-8B) models. Experiments are conducted on two NVIDIA Ada RTX 6000 with 48 GBs of memory each.

4.1 Language Modeling Performance

Table 1 compares perplexity (PPL) for LaMoC against the aforementioned joint compression baselines given compression rates of 20%, 30%, 40%, and 50%. We draw the following observations:

Overall Performance. Across all models and compression rates, LaMoC with gradient-error alignment improves the language modeling for all models across the different compression settings. The relative reduction in PPL compared to MoDeGPT reaches an average of 2.46% for a relative reduction range of 0.79% – 6.79%. The average absolute perplexity reduction is at 0.8 with an absolute PPL reduction range of 0.07-6.11.

Model Type Influence. The choice of model family exhibits an effect on the degree of improvement. Taking the 8B tier as an example, the relative reduction in PPL of Llama-3.1-8B is at 1.16% (0.19 abs) contrary to Ministral-8B-It at 2.82% (0.39), suggesting how original training mechanics and task prioritization could have an impact.

Model Size Impact. Smaller models are more likely to benefit from the gradient-alignment approach to recover loss. The Qwen3-4B-It-2507 achieves a better average relative reduction of 3.39% with an average absolute PPL reduction of 1.83. We verify this observation on the smaller Llama-3.2-1B and Llama-3.2-3B in Table 2, where the average reduction can reach 3.20% for a fair subset of the compression ratios.

Compression Rate Impact. Gradient alignment benefits are more observable at aggressive compression rates, which can be attributed to the increasing loss scale. The scale of absolute PPL reduction increases as the compression rate increases – from 0.23 at 20% to 2.73 at 50%.

Choice of γ Strategy. The adaptive strategy offers the strongest relative reduction in PPL (2.42%) compared to the fixed options (1.42% and 1.52%), respectively. For each configuration compared to the best fixed strategy, the adaptive γ strategy leads to PPL reduction reaching up to 5.4 points compared to the best fixed γ option.

4.2 Task Proficiency Performance

Table 3 showcases the downstream task performance for the 5-shot MMLU and 0-shot tasks listed beforehand. We compare the performance at 20% and 40% for the Llama-3.1-8B, Ministral-8B-It-2410, and Qwen3-4B-It-2507 in the following.

Overall Performance. Both fixed and adaptive modes of LaMoC improve over the baseline MoDeGPT. The improvements reach on average across all models and compression rates an increase of +0.99% (+0.53 pp) across all benchmarks, broken

[2]We use Qwen3-4B-Instruct-2507, Qwen3-4B-Inst., Qwen3-4B-It, and Qwen3-4B interchangeably in this work

Config	W	T	L	Mean acc. improv.
Fixed $\gamma = 0.75$	21	2	13	$+0.41 \pm 1.07$ pp
Adaptive $\gamma^\star$	20	0	16	$+0.40 \pm 1.11$ pp
Best	**26**	**1**	**9**	$\mathbf{+0.67 \pm 1.11}$ **pp**

Table 4: Win/tie/loss counts and mean task accuracy improvement based on the data in Table 3.

	Ministral-8B		Qwen3-4B		Qwen3-8B	
	γ=1	$\gamma^\star$	γ=1	$\gamma^\star$	γ=1	$\gamma^\star$
PPL ↓	12.16	**11.95**	16.75	**16.08**	15.70	**15.10**
0-shot Avg ↑	69.18	**69.43**	64.44	**64.94**	65.78	**66.53**
MMLU ↑	58.80	**59.17**	53.60	**56.13**	59.46	**60.20**

Table 5: Calibration data ablation via Alpaca (Taori et al., 2023) at 20% compression on instruction models.

	Fixed $\gamma = 0.75$		**Adaptive** $\gamma^\star$	
	EF	**GGN**	**EF**	**GGN**
PPL ↓	41.70	41.46	41.29	41.05
LM-Avg ↑	41.75	41.74	42.12	41.54

Table 6: Loss ablation at 40% comp. for Llama-3.2-1B.

Model	Ratio	Base	Trace ✓	Trace ×
Llama-3.2-1B	0.8	17.52	**17.21**	17.52
	0.6	44.21	**41.29**	44.04
Qwen3-4B-It	0.8	12.28	**12.02**	12.28
	0.6	25.79	**25.24**	25.69

Table 7: Trace-rescaling ablation: WikiText-2 PPL (↓).

down to +3.89% (+1.65 pp) on the 5-shot MMLU, and +0.71% (+0.40 pp) for the 0-shot accuracy.

Statistical Analysis. In Table 4, we show the Win/Tie/Loss stats and average task accuracy improvement compared to the baseline based on the task accuracy data for the 6 benchmarks from Table 3. We observe that both Fixed $\gamma = 0.75$ and Adaptive $\gamma^\star$ implementations achieve improvements over the baseline, with the best strategy choice achieving $+0.67 \pm 1.11$ pp accuracy improvement.

Model Effect. The smaller Qwen3-4B-It-2507 experiences the largest improvement +1.89% (+0.98) with MMLU reaching +6.30% (+2.24 pp), compared to larger models like Llama-3.1-8B which improves by +3.65% (+1.78 pp). This suggests how model type and size affects the degree of benefiting from gradient loss alignment given the varying information packing density per parameter translating into different scale of loss effects.

Compression Rate Effect. We also observe the improvements for the 40% compression rate reaches +1.14% (+0.52 pp) compared to the 20% rate achieving +0.89% (+0.54 pp). This goes in line with the insight that the benefits from gradient-loss aligned compression rise as the loss increases.

4.3 Ablation

Per-Module Contribution. Figure 4 demonstrates performance contributions from LaMoC across the different modules from Qwen3-4B-It-2507 at 20% and 40% compression. We find that LaMoC application on MLP + VO offers the largest positive change in performance on both evaluation suites.

Calibration Data. In Table 5, we ablate the choice of compression calibration dataset by using 128 samples from the Alpaca instruction dataset (Taori et al., 2023) on the instruction-tuned models from the aforementioned list of models. Two observations: (1) Adaptive $\gamma^\star$ wins on all 6 downstream tasks with average accuracy improvement of $+0.86 \pm 0.84$ pp compared to the baseline while being consistent across benchmarks – overall improvements reach 3.32%, 0.75%, and 2.12% for perplexity, 0-shot average and MMLU, respectively. (2) Using Alpaca dataset for calibration yields a trade-off in language modeling and task accuracy, consistent with prior works' observations.

Loss Signal Ablation. We ablate the loss signal used to bias the effective Gram, comparing Empirical Fisher (EF) against generalized Gauss–Newton (GGN) (Botev et al., 2017). Table 6 reports results on Llama-3.2-1B under fixed $\gamma = 0.75$ and adaptive $\gamma^\star$. We observe GGN trading off perplexity gains for LM-Avg reduction, seen in the adaptive $\gamma^\star$ setting where PPL is reduced by 0.24, while LM-Avg is reduced by 0.58 points relative to EF.

Trace Rescaling Ablation. We ablate the effect of trace rescaling from Equation 4 by invoking compression with Adaptive $\gamma^\star$ with and without trace rescaling. Tables 7 and 8 show how the results change for WikiText-2 and average 0-shot accuracy for Llama-3.2-1B and Qwen3-4B-It. The results demonstrate the importance of trace rescaling to our method, where on average with trace rescaling, perplexity improvements rise from 0.19% to 3.16% while task accuracy gains rise from 0.02 to 0.40 pp.

Scalability. We further assess the scalability of LaMoC by extending evaluations to Qwen2.5-32B (Qwen Team, 2024) using 128 samples from WikiText-2 for calibration. For this tier of models, we use UniQL as our modular compression base-

Model	Ratio	Base	Trace ✓	Trace ×
Llama-3.2-1B	0.8	49.94	49.15	**49.95**
	0.6	41.34	**42.12**	41.36
Qwen3-4B-It	0.8	62.97	**63.82**	62.98
	0.6	47.10	**47.88**	47.13

Table 8: Trace-rescaling ablation: 0-shot average (↑).

Rate	Config	WikiT-2 PPL ↓	0-shot Avg ↑
20%	UniQL	6.73	69.74
	Adaptive $\gamma^\star$	**6.68**	**71.16**
40%	UniQL	10.58	54.00
	Adaptive $\gamma^\star$	**10.42**	**55.25**

Table 9: Qwen2.5-32B evaluations post compression.

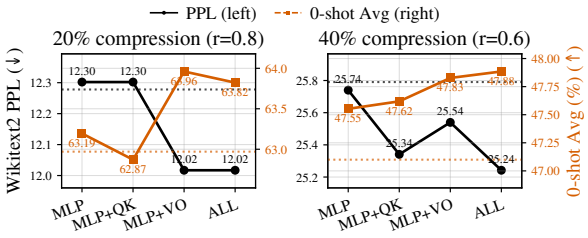


Figure 4: Per-module performance improvements across the different Qwen3-4B-It-2507 modules.

line, and perform our evaluations on an NVIDIA RTX PRO 6000 Blackwell with 96 GB of memory. Table 9 shows the results for 20% and 40% compression. We observe on average $+1.34$ pp improvements in the 0-shot average task accuracy. In Appendix C.2, we show further scalability evaluations on EXAONE 4.5-33B (Choi et al., 2026).

Repeated Sampling. In Table 10, we repeat our evaluation on the Qwen3-4B-Instruct-2507 at 20% and 40% using 3 different calibration sampling seeds to ensure consistency of the performance gains. We find that 0-shot accuracy improvement holds at $+0.76 \pm 0.26$ pp.

4.4 Latency Performance Benchmarking

We benchmark the end-to-end latency on the NVIDIA RTX Ada 6000 using `torch.compile`, setting batch size of 1 and 2048 sequence length. For prefill, we take the median of 20 runs, whereas for decode, we use greedy decode and take the median time per output token for 64 decode tokens. In Table 11, we show the results of the benchmarking at 20% compression for the Llama-3.1-8B and the Qwen3-4B-Inst. On average, both models achieve $1.1\times$ speedup compared to their dense baselines.

Rate	Config	WikiT-2 PPL ↓	0-shot Avg ↑
20%	Base	12.29 ± 0.01	63.50 ± 0.46
	Adaptive $\gamma^\star$	$\mathbf{11.99 \pm 0.06}$	$\mathbf{64.18 \pm 0.38}$
40%	Base	25.59 ± 0.18	46.76 ± 0.61
	Adaptive $\gamma^\star$	$\mathbf{24.96 \pm 0.25}$	$\mathbf{47.60 \pm 0.35}$

Table 10: Repeated Qwen3-4B-Instruct-2507 evaluations under 3 different seeds of calibration data.

	Llama-3.1-8B		**Qwen3-4B**	
	Dense	20%	Dense	20%
Prefill (ms)	341.1	**278.9**	185.0	**165.2**
Decode (ms/tok)	21.91	**20.2**	14.8	**13.7**
Speedup(N=64)	1.0×	**1.1**×	1.0×	**1.1**×

Table 11: End-to-end inference latency at 20% modular compression via `torch.compile`.

5 Discussion

Below are our key insights captured from gradient-loss information inclusion in modular compression.

Performance gains stack. We motivated this work by discussing the four key ideas in SOTA training-free compression frameworks: Activation-awareness, loss curvature information, non-uniform truncation, and modularity. This work demonstrated that the stacking of the four techniques can unlock further performance gains.

Generality. LaMoC's proposal in modifying the activation statistics themselves makes it complementary to the techniques employed by any compression framework. LaMoC can scale to any framework that relies on activation statistics.

Future Directions. LaMoC is based on empirical analysis and statistical observations, which limits our exploration to a small subset of candidate γ values and a heuristic γ selection method. Deriving an analytical method to solve for optimal γ per each module can be investigated in future.

6 Conclusion

We provided a characterization for loss-aware modular compression and presented LaMoC, a gradient-error aligned methodology for modular LLM compression. LaMoC injects loss sensitivity through gradient-weighted Gram matrices to address the two-tiered problem formulation over module reconstruction and gradient information blending rate. Our experiments have shown LaMoC improving over state-of-the-art modular compression methods across benchmarks and compression rates.

Limitations

Our proposed methodology is derived based on empirical observations on the effects of incorporating gradient information onto the joint compression objective. Deriving a theoretically-proven analytical solution for γ remains important for future exploration due to its potential for unlocking further performance gains and improving compression speed. Our method has been tested and evaluated for models ranging from 1B to 33B in model size. Still, broader experimental validation at or beyond the $\geq 30B$ parameter tier of models remains needed. Furthermore, evaluating the effectiveness of this methodology on emerging model architectures (Hybrid, State-space models, linear attention, Mixture of Experts) and tasks (long context reasoning, agentic tool calling) is a focus for future investigations.

References

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. 2024. SliceGPT: Compress large language models by deleting rows and columns. In *The Twelfth International Conference on Learning Representations*.

Guangji Bai, Yijiang Li, Chen Ling, Kibaek Kim, and Liang Zhao. 2024. Sparsellm: Towards global pruning of pre-trained language models. *Advances in Neural Information Processing Systems*, 37:46203–46225.

Aleksandar Botev, Hippolyt Ritter, and David Barber. 2017. Practical gauss-newton optimisation for deep learning. In *International Conference on Machine Learning*, pages 557–565. PMLR.

Hung-Yueh Chiang, Chi-Chih Chang, Yu-Chen Lu, Chien-Yu Lin, Kai-Chiang Wu, Mohamed S. Abdelfattah, and Diana Marculescu. 2026. UniQL: Unified quantization and low-rank compression for adaptive edge LLMs. In *The Fourteenth International Conference on Learning Representations*.

Eunbi Choi, Kibong Choi, Sehyun Chun, Seokhee Hong, Junwon Hwang, Hyojin Jeon, Ahra Jo, Hyunjik Jo, Yeonsik Jo, Joonkee Kim, Seonghwan Kim, Soyeon Kim, Sunkyoung Kim, Yireun Kim, Yongil Kim, Changhun Lee, Haeju Lee, Jinsik Lee, Kyungmin Lee, and 39 others. 2026. Exaone 4.5 technical report. *Preprint*, arXiv:2604.08644.

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

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

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2023. A framework for few-shot language model evaluation.

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

Babak Hassibi, David G Stork, and Gregory J Wolff. 1993. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, pages 293–299. IEEE.

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 Representations*.

Xing Hu, Dawei Yang, Yuan Cheng, Zhixuan Chen, and Zukang Xu. 2026. SAES-SVD: Self-adaptive suppression of accumulated and local errors for SVD-based LLM compression. In *The Fourteenth International Conference on Learning Representations*.

Toshiaki Koike-Akino, Xiangyu Chen, Jing Liu, Ye Wang, Pu, Wang, and Matthew Brand. 2025. Latentllm: Attention-aware joint tensor compression. *Preprint*, arXiv:2505.18413.

Yann LeCun, John Denker, and Sara Solla. 1989. Optimal brain damage. *Advances in neural information processing systems*, 2.

Zhiteng Li, Mingyuan Xia, Jingyuan Zhang, Zheng Hui, Haotong Qin, Linghe Kong, Yulun Zhang, and Xiaokang Yang. 2025. Adasvd: Adaptive singular value decomposition for large language models. *arXiv preprint arXiv:2502.01403*.

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 *The Thirteenth International Conference on Learning Representations*.

James Liu, Pragaash Ponnusamy, Tianle Cai, Han Guo, Yoon Kim, and Ben Athiwaratkun. 2025. Training-free activation sparsity in large language models. In *The Thirteenth International Conference on Learning Representations*.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. LLM-pruner: On the structural pruning of large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Xin Men, Mingyu Xu, Qingyu Zhang, Qianhao Yuan, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. 2025. Shortgpt: Layers in large language models are more redundant than you expect. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20192–20204.

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

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

Mistral AI. 2024. Ministral-8b-instruct-2410. https://huggingface.co/mistralai/Ministral-8B-Instruct-2410. Released October 2024.

Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. 2019. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11264–11272.

Mohanad Odema, Gabrielle De Micheli, Dayin Gou, Nilesh Malpeddi, Prathamesh Vaste, and Jacob Song. 2026. Understanding calibration and truncation error propagation in training-free low-rank compression for llms. *arXiv preprint arXiv:2608.08506*.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Ayan Sengupta, Siddhant Chaudhary, and Tanmoy Chakraborty. 2025. You only prune once: Designing calibration-free model compression with policy learning. In *The Thirteenth International Conference on Learning Representations*.

Sidak Pal Singh and Dan Alistarh. 2020. Woodfisher: Efficient second-order approximation for neural network compression. *Advances in Neural Information Processing Systems*, 33:18098–18109.

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

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Tycho F. A. van der Ouderaa, Markus Nagel, Mart Van Baalen, and Tijmen Blankevoort. 2024. The LLM surgeon. In *The Twelfth International Conference on Learning Representations*.

Qinsi Wang, Jinghan Ke, Masayoshi Tomizuka, Kurt Keutzer, and Chenfeng Xu. 2025a. Dobi-svd: Differentiable svd for llm compression and some new perspectives. In *International Conference on Learning Representations*, volume 2025, pages 12561–12590.

Xin Wang, Samiul Alam, Zhongwei Wan, Hui Shen, and Mi Zhang. 2025b. Svd-llm v2: Optimizing singular value truncation for large language model compression. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4287–4296.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. 2025c. SVD-LLM: Truncation-aware singular value decomposition for large language model compression. In *The Thirteenth International Conference on Learning Representations*.

Jeffrey TH Wong, Cheng Zhang, Xinye Cao, Pedro Gimenes, George A Constantinides, Wayne Luk, and Yiren Zhao. 2025. A3: an analytical low-rank approximation framework for attention. *arXiv preprint arXiv:2505.12942*.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1513–1528.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Mingzhe Yang, Sihao Lin, Changlin Li, and Xiaojun Chang. 2025b. Let llm tell what to prune and how much to prune. In *Forty-second International Conference on Machine Learning*.

Ruokai Yin, Yuhang Li, Donghyun Lee, and Priyadarshini Panda. 2025. DuoGPT: Training-free dual sparsity through activation-aware pruning in LLMs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Zhihang Yuan, Yuzhang Shang, Yue Song, Dawei Yang, Qiang Wu, Yan Yan, and Guangyu Sun. 2023. Asvd: Activation-aware singular value decomposition for compressing large language models. *arXiv preprint arXiv:2312.05821*.

A AI Usage

The original ideas, methodology design, and experimental planning in the paper are from the authors. We use AI assistants in coding, writing assistance, formatting, and experimental design refinement.

B Detailed Related Works Positioning

We further elaborate on the relation of this work to existing related works mentioned in Section 2.1.

B.1 Comparison to loss-aware approaches

Table 12 provides a comprehensive comparison comparing LaMoC against existing works that consider Hessian/Fisher/gradient-based approaches to compression or pruning. LaMoC's key contribution lies in the formulation of loss-aware low-rank compression in the module activation space to blend gradient-weighted and canonical activation statistics. From there, LaMoC proposes heuristic-based fixed and adaptive strategies to control the blending degree, where the adaptive strategy uses gradient-error alignment proxy to predict the loss effect of each candidate reconstruction.

B.2 Modular compression landscape

The majority of our experiments implement LaMoC on top of MoDeGPT (Lin et al., 2025) or UniQL (Chiang et al., 2026) owing to them being state-of-the-art in modular compression. These works outperform conventional single layer low-rank approximation or pruning approaches. For instance, Table 13 is from UniQL showing its superiority in compressing Llama-3.1-8B at 25% rate compared to existing pruning or single matrix low-rank compression approaches, where UniQL maintains better average 0-shot accuracy post compression without any finetuning requirement.

Additional frameworks that belong to the same MoDeGPT-style modular compression family include A3 (Wong et al., 2025) and LatentLLM (Koike-Akino et al., 2025). UniQL was prioritized in this work based on the reported results of related works. The closest work was A3, where both UniQL and A3 shared a reporting on 5-shot MMLU for Llama-3.1-8B where UniQL achieved 60.2% at 15% compression compared to A3 which achieved 59.22% at 10% compression.

C Additional Experiments

C.1 Calibration effects

We evaluate the effects of changing the calibration dataset using the EXAONE 4.5-33B (Choi et al., 2026) compared to the baseline UniQL. Specifically, we construct an alternative calibration data mix to collect the empirical-Fisher gradient grams, whereas the input covariance and layer ratio allocation remain calibrated through WikiText-2. The new calibration mix constitutes 128 samples distributed equally — 25% Evol-CodeAlpaca (code), 25% ClimbMix (English web text), 25% Orca-Math English CoT (reasoning), and 25% Korean (a seven-source in-house mix). The results at 20% compression are shown in Table 14, where we observe a slight capability trade-off between perplexity and downstream task proficiency. In either case, Adaptive $\gamma^\star$ remains superior to the baseline.

C.2 Aggressive Compression.

We evaluate aggressive compression across two tiers of models as follows:

Instruct Models (50%). We evaluate the Adaptive $\gamma^\star$ at 50% compression for the Qwen3-4B-Instruct and Ministral-8B models in Table 16 compared to the baseline MoDeGPT (Lin et al., 2025). We observe the Adaptive $\gamma^\star$ still outperforms the baseline in perplexity and 0-shot accuracy.

30B model tier (60%-70%). We evaluate our Adaptive $\gamma^\star$ approach under 60% and 70% for EXAONE 4.5-33B (Choi et al., 2026) compared to the baseline UniQL (Chiang et al., 2026) modular compression approach. The results are shown in Table 17. We observe at both compression rates, the Adaptive $\gamma^\star$ improves both the WikiText-2 and 0-shot accuracy compared to the baseline UniQL.

C.3 Adaptive γ Selection

We study the viability of adaptive γ as a heuristic-driven solution for the two-tiered problem formulation. From Section 3.3, we first perform a detailed analysis on the selection proxy choices in Figure 3 against an Oracle that is aware of the best γ configuration per layer.

Generalized Adaptive $\gamma^\star$ formulation. We define a more generalized form of the adaptive $\gamma^\star$ selection policy in Equation 9 with a confidence constrained blending condition incorporated as:

Method	Loss approximation	Key contribution	Relation to LaMoC
OBD (LeCun et al., 1989)	Second-order	Introduces diagonal-Hessian saliency for loss-aware weight pruning	Establishes per-weight loss-aware compression
OBS (Hassibi et al., 1993)	Second-order	Introduces inverse-Hessian pruning and weight loss compensation	Models weight interactions for pruning compensation
WoodFisher (Singh and Alistarh, 2020)	Second-order; optional first-order term	Makes OBS-style pruning practical for larger networks by approximating the inverse empirical Fisher	Makes second-order pruning scalable
LLM Surgeon (van der Ouderaa et al., 2024)	Second-order; optional first-order correction	Extends OBS-style pruning to LLMs using layer-wise activation and gradient statistics	Weights or dimensions pruning; LaMoC operates on module level
LLM-Pruner (Ma et al., 2023)	First- and approximate second-order importance	Finds connected structures to be removed together, ranked by loss effect	Prunes module structures; LaMoC performs module-level low-rank approximation
FWSVD (Hsu et al., 2022)	Diagonal empirical-Fisher-weighted reconstruction	Introduces row-wise Fisher-weighted low-rank approximation	Fixed row-wise Fisher weighting in weight space vs. adaptive activation–loss blending in module activation space

Table 12: Positioning of LaMoC in relation to loss-aware compression works.

Method	Avg. 0-shot acc. ↑
ShortGPT (Men et al., 2025)	43.24
SliceGPT (Ashkboos et al., 2024)	46.17
SVD-LLM (Wang et al., 2025c)	54.20
UniQL (Chiang et al., 2026)	**66.56**

Table 13: Comparison against single-matrix compression and pruning from UniQL (Chiang et al., 2026)

Config	WikiT-2 PPL ↓	0-shot Avg ↑
UniQL ($\gamma = 1.0$)	12.97	69.87
Adaptive $\gamma^\star$ (Wiki)	11.81	**70.37**
Adaptive $\gamma^\star$ (Mix)	**11.62**	70.14

Table 14: Comparing EXAONE 4.5-33B results under 128 calibration samples from WikiT-2 and datamix (coding, Climbmix, reasoning) at 20% compression

$$\gamma_m^\star = \begin{cases} \arg\min_{\gamma_m \in \Gamma_m} \mathbf{p}(\gamma_m), & \text{if } \min_{\gamma_m \in \Gamma_m} \mathbf{p}(\gamma_m) \\ & < -\varepsilon - k\,\mathrm{SE}(\gamma_m), \\ 1, & \text{otherwise.} \end{cases} \tag{10}$$

where ε and k are constants; SE represents the standard error of the per-sequence proxy values across the calibration set, defined as:

$$\mathrm{SE}(\gamma_m) = \frac{\mathrm{sd}(\{p_i(\gamma_m)\}_{i=1}^n)}{\sqrt{n}}, \tag{11}$$

where n is the number of calibration data samples (here we use n=128 from Alpaca), sd is the standard deviation over the n per-sequence proxy values $p_i(\gamma_m)$.

The generalized Equation 10 applies the confidence threshold per candidate, where the candidate with the lowest predicted cross-entropy change among the valid options is assigned to $\gamma_m^\star$. This formulation allows blending only when the predicted cross-entropy reduction is (1) statistically significant, at least k standard errors below zero, and (2) of non-negligible magnitude, exceeding ε. In other words, Equation 9 is a special case at $\varepsilon = 0, k = 0$.

γ choices analysis vs. Oracle. In Table 15, we showcase the comparison of Adaptive $\gamma^\star$ selection approaches against the Oracle. From the table, we observe (1) the proxy can be more aggressive in picking γ for gradient blending; (2) A more constrained implementation can better match the oracle

Config	#Blends (max 16)	Oracle γ agree	WikiT-2 PPL ↓	0-shot Avg ↑
Baseline ($\gamma = 1$)	0	N/A	19.65	54.98
Adaptive $\gamma^\star$ ($\varepsilon = 0, k = 0$)	14	7	17.31	54.60
Adaptive $\gamma^\star$ ($\varepsilon = 10^{-6}, k = 3$)	9	11	17.52	54.25
Oracle	9	16	17.53	54.68

Table 15: Adaptive $\gamma^\star$ selection for Llama-3.2-1B MLP modules at $r = 0.85$ compared to Oracle selection.

Model	Config	WikiT-2 ↓	0-shot Avg ↑
Qwen3-4B	MoDeGPT	90.02	41.25
	Adaptive $\gamma^\star$	**83.91**	**41.76**
Ministral-8B	MoDeGPT	20.60	44.76
	Adaptive $\gamma^\star$	**20.05**	**44.99**

Table 16: Instruction-tuned models 50% compression with Adaptive $\gamma^\star$ compared to MoDeGPT ($\gamma = 1.0$).

r	Config	WikiT-2 PPL ↓	0-shot Avg ↑
0.8	UniQL ($\gamma = 1.0$)	12.97	69.87
	Adaptive $\gamma^\star$	**11.62**	**70.14**
0.4	UniQL ($\gamma = 1.0$)	471.7	40.13
	Adaptive $\gamma^\star$	**356.0**	**42.59**
0.3	UniQL ($\gamma = 1.0$)	1498.9	37.68
	Adaptive $\gamma^\star$	**1030.2**	**38.86**

Table 17: EXAONE 4.5-33B compression with Adaptive $\gamma^\star$ with focus on aggressive compression rates.

choice, though not necessarily improve the performance; (3) Per-layer optimal γ selection does not directly translate to pure downstream gains given other effects as compounded selection.

Adaptive γ Selection Statistics. We report the γ selection choices for EXAONE 4.5-33B and Qwen3-4B-Instruct where both outperform their respective baselines on the target tasks. For the EXAONE, we demonstrate the choices when using the WikiText-2 and datamix as the empirical Fisher calibration datasets. The results are displayed in Tables 18 and 19, where we observe that the blending option offered through LaMoC is selected 70%-77% of the time across both tables.

C.4 Free-form Generation Evaluation

We further evaluate the EXAONE 4.5-33B on free-form generation benchmarks (GSM8K, HumanEval, MT-Bench) shown in Table 21 at 20% compression compared to the UniQL baseline. We observe that the Adaptive $\gamma^\star$ outperforms the UniQL on the three benchmarks.

EXAONE	MLP	VO	QK
Wiki (r=0.8)	51 | 20 | 31	42 | 15 | 27	42 | 19 | 23
Mix (r=0.8)	51 | 28 | 23	37 | 7 | 30	43 | 18 | 25
Mix (r=0.4)	46 | 26 | 20	40 | 9 | 31	48 | 12 | 36

Table 18: γ selection counts for EXAONE 4.5-33B (64 layers) at different calibration and compression. Entry format xlylz reflects $\gamma < 1.0$ | γ=0.75 | γ=0.5 choices.

Qwen3-4B-It	MLP	VO	QK
Wiki (r=0.8)	22 | 14 | 8	33 | 12 | 21	23 | 9 | 14
Wiki (r=0.6)	32 | 16 | 16	33 | 14 | 19	24 | 8 | 16

Table 19: γ selection counts for Qwen3-4B-Inst. (36 layers) at different calibration and compression. Entry format xlylz reflects $\gamma < 1.0$ | γ=0.75 | γ=0.5 choices.

C.5 Candidate Selection Set Granularity

We investigate the value from including additional γ candidates into the γ selection process. We expand the Adaptive γ grid to include $\{0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 1.0\}$. We repeat the experiment for WikiT-2 and Alpaca calibrations on Qwen3-4B-Inst. at 20% compression in Tables 22 and 23. The key observations are: (1) Finer or coarser grids can both lead to accuracy gains against the baseline MoDeGPT; (2) The finer grid can be affected by predictor noise, coarser grid introduces generalization (see Table 23).

C.6 Reference Gradients Ablation

We ablate the effect of using approximate reference gradients, computed once on the canonical ($\gamma = 1.0$) compressed model and reused across all candidates, against exact gradients recomputed for each candidate, where the candidate's compressed weights are applied at its layer leaving all other layers in the canonical ($\gamma = 1.0$) state. As shown in Table 24, reference gradients maintain performance close to that of exact gradients while requiring $37.9\times$ less backward-pass compute.

C.7 Compression Times

In Table 25, we showcase the compression times taken by MoDeGPT and LaMoC when running

Statistic	MLP	VO	QK
Gradient Gram G (per layer)	[hidden, hidden]	[hidden, hidden]	Q: $[n_q \cdot \mathrm{head}_{\mathrm{dim}},\ n_q \cdot \mathrm{head}_{\mathrm{dim}}]$ K: $[n_{\mathrm{kv}} \cdot \mathrm{head}_{\mathrm{dim}},\ n_{\mathrm{kv}} \cdot \mathrm{head}_{\mathrm{dim}}]$
Fisher diagonal w (VO: per KV group; QK: per head)	$[d_{\mathrm{int}}]$	$[\mathrm{head}_{\mathrm{dim}}]$	$[\mathrm{head}_{\mathrm{dim}}]$
Activation Gram $\mathcal{C}_0$, Fisher-weighted Gram $\mathcal{C}_F$ (VO: per KV group; QK: per head)	$[d_{\mathrm{int}},\ d_{\mathrm{int}}]$	$[\mathrm{head}_{\mathrm{dim}},\ \mathrm{head}_{\mathrm{dim}}]$	$[\mathrm{head}_{\mathrm{dim}},\ \mathrm{head}_{\mathrm{dim}}]$

Table 20: Tensor shapes per module. G is collected once per layer at the module's output projection; $w = \mathrm{diag}(W^\top G W)$ carries it back into the activation space (for QK w is simply the diagonal of G), where $\mathcal{C}_F = \mathrm{diag}(\sqrt{w})\,\mathcal{C}_0\,\mathrm{diag}(\sqrt{w})$ shares $\mathcal{C}_0$'s shape by construction. VO statistics are formed per KV group (n_{kv} per layer); QK statistics are per head.

Config	GSM8K↑	HumanEval↑	MT-Bench↑
UniQL	38.2%	9.1%	2.7%
Adaptive $\gamma^\star$	**39.6%**	**14.6%**	**3.7%**

Table 21: Adaptive $\gamma^\star$ evaluations at $r = 0.8$ compared to UniQL ($\gamma = 1.0$) for free-form generation.

Config (wiki calibration)	PPL ↓	0-shot Avg ↑
Base ($\gamma = 1.0$)	12.28	62.97
Adaptive $\gamma^\star$ (coarse)	12.02	**63.82**
Adaptive $\gamma^\star$ (fine)	**11.83**	63.69

Table 22: Candidate γ set granularity ablation, Qwen3-4B-Inst at 20% compression, wiki calibration.

their respective compression methods on the NVIDIA RTX Ada 6000 machine. LaMoC's additional compression time (2.5 h vs 3 h 10 min for Llama-3.1-8B) comes mainly from having to compute and store the gradients of the reference solution, and having to assess every candidate in the compression pipeline. The increase in time remains manageable with potential for further future optimization to target speeding up the operation.

D Tensor Shapes

In Table 20, we provide the tensor shapes for the key data structures: the Gradient Gram G, the Fisher diagonal w, and the Activation and Fisher-weighted activation Grams ($\mathcal{C}_0$, $\mathcal{C}_F$) across MLP, VO, and QK modules. hidden is the model hidden dimension; $\mathrm{head}_{\mathrm{dim}}$ is the head dimension; d_{int} is the MLP intermediate dimension; n_q and n_{kv} are the numbers of query and KV heads, respectively.

Config (alpaca calibration)	PPL ↓	0-shot Avg ↑
Base ($\gamma = 1.0$)	16.75	64.41
Adaptive $\gamma^\star$ (coarse)	**16.08**	**64.91**
Adaptive $\gamma^\star$ (fine)	17.69	64.51

Table 23: Candidate γ set granularity ablation, Qwen3-4B-Inst at 20% compression, alpaca calibration.

Gradient	Relative cost	PPL ↓	0-shot Avg ↑
Baseline	–	12.28	62.97
Ref. gradient	1×	12.30	**63.19**
Exact gradient	37.9×	**12.28**	63.12

Table 24: Reference vs. exact gradients on Qwen3-4B-Instruct at $r = 0.8$ with wiki calibration (MLP modules). Relative cost is the number of backward passes normalized to the reference-gradient setting.

	Qwen3-4B	**Llama-3.1-8B**
MoDeGPT canonical	32 min	∼2.5 h
LaMoC: grads cache	+2 min	+3 min
LaMoC: candidates	+24 min	+35 min
Total LaMoC	**∼58 min**	**∼3 h 10 min**

Table 25: Wall time of LaMoC compression compared to canonical MoDeGPT at 20% rate on NVIDIA RTX Ada 6000. LaMoC consumes additional time in: (1) Gradients caching from the canonical reference; (2) Evaluating the candidates in each module depending on the number of layers, γ choices and number of modules. Analysis using 128 samples from WikiText-2.