LaCo: Layer-wise Compensation for Pruned Large Language Models

Yingen Liu, Fan Wu*, Xuyan Pan, Ruihui Li, Zhuo Tang*, Kenli Li
College of Computer Science and Electronic Engineering, Hunan University
{liuyingen, wufan, panxy2005, liruihui, ztang, lkl}@hnu.edu.cn

Abstract

Pruning is essential for the efficient deployment of Large Language Models (LLMs); however, it causes severe performance degradation due to the structural distortion induced by sparsity. Existing recovery strategies, such as LoRA, predominantly employ global fine-tuning, often overlooking the mechanistic root of this degradation: the layer-wise accumulation and amplification of local errors. To address this limitation, we propose **LaCo** (**La**yer-wise **Co**mpensation), a framework that reorients the recovery paradigm from global adaptation to hierarchical representation alignment. By sequentially optimizing each layer to reconstruct the model's hidden states, LaCo effectively intercepts the error propagation chain at its source. Extensive experiments demonstrate that LaCo surpasses parameter-efficient baselines in both perplexity reduction and zero-shot reasoning. Notably, it reduces recovery-time memory usage to approximately $1/7$ of the baseline and requires only 2,048 unlabeled samples to match a LoRA model trained on 50k examples—achieving a $\sim 25\times$ improvement in data efficiency.

1 Introduction

Pruning has emerged as a pivotal strategy to alleviate the substantial memory and latency burdens of Large Language Models (LLMs) (OpenAI et al., 2024; Team et al., 2023; Cheng et al., 2024). While recent weight removal techniques—such as LLM-Pruner (Ma et al., 2023), Wanda (Sun et al., 2023), and SparseGPT (Frantar and Alistarh, 2023)—effectively discard parameters to accelerate inference, they primarily optimize for sparsification efficiency, **thereby neglecting the core challenge of post-pruning performance recovery**. Since sparsification inevitably compromises the model's representational integrity, effective post-pruning compensation is not merely an optional enhancement but the decisive factor for deployment. However, the field currently lacks a robust recovery method that effectively restores capabilities while maintaining resource efficiency.

We argue that principled post-pruning recovery should aim to repair the internal representations disrupted by sparsity, ensuring the pruned model's internal behavior aligns with the original model. An ideal recovery strategy should thus fulfill three criteria: (1) **Generality**, being agnostic to the pruning paradigm; (2) **Efficiency**, requiring minimal data and computation; and (3) **Fidelity**, directly targeting the structural distortion without injecting external task bias.

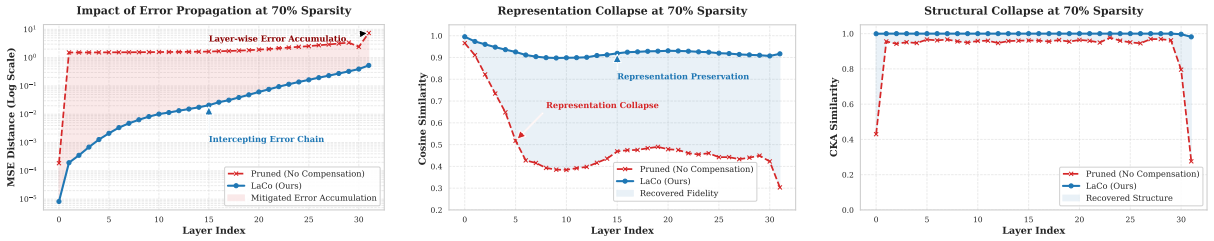
Mainstream parameter-efficient approaches adapted for recovery, such as LoRA(Hu et al., 2022), struggle to satisfy these criteria simultaneously. By relying on global fine-tuning with downstream data, such methods prioritize task-specific adaptation over generic representation restoration. This approach constitutes an indirect and task-biased compensation, overlooking the mechanistic root of degradation. Fundamentally, they treat the **symptom** (poor task performance) rather than the **disease** (structural distortion).

Our analysis pinpoints the **disease**: degradation stems not from end-to-end prediction error, but from the layer-wise accumulation and amplification of local errors(Sec. 3.1). Each layer's output discrepancy is propagated and magnified by subsequent layers, leading to catastrophic performance collapse. This yields a critical hypothesis: **effective recovery necessitates rectifying the error propagation at its source.**

Guided by this insight, we propose **LaCo** (**La**yer-wise **Co**mpensation), which redefines recovery through hierarchical representation alignment. LaCo decomposes the intractable global problem into tractable, local regression subproblems, optimizing each sparse layer to reconstruct the original model's output hidden states.

* Corresponding authors.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 29099–29113



(a) **Error Accumulation** (b) **Cosine Similarity** (c) **CKA Similarity**

Figure 1: **Mechanistic visualization of LaCo's impact at 70% sparsity.** (a) Pruning induces layer-wise error accumulation, which LaCo effectively severs. (b) Semantic Consistency: Cosine similarity drops significantly in pruned models (Red), indicating representation collapse, while LaCo (Blue) maintains high fidelity. (c) Structural Consistency: CKA analysis further confirms that LaCo preserves the internal feature structure better than the baseline.

Crucially, its interleaved calibrate-then-propagate strategy—using each layer's compensated output as the input for the next layer—curtails the error propagation chain at its source. This design inherently ensures fidelity (direct representation repair), efficiency (single-layer optimization), and generality (agnostic to the pruning pattern).

Comprehensive experiments across diverse pruning paradigms and models confirm that LaCo consistently outperforms strong baselines like LoRA, with its advantage most pronounced under extreme sparsity (e.g., 70%). Notably, LaCo achieves this superior recovery with drastic resource efficiency: it cuts peak memory overhead by over **7**× and matches baseline performance using only ∼2,000 unlabeled samples—a **25**× **reduction in data requirements**. This efficiency, coupled with its versatility as both an on-the-fly enhancement and a post-pruning compensation strategy, establishes LaCo as a practical recovery paradigm.

Our main contributions are as follows:

- **Theoretical Insight:** We identify layer-wise error accumulation as the mechanistic root of pruning degradation, establishing the theoretical basis for a paradigm shift from global adaptation to hierarchical representation alignment.
- **Methodological Innovation:** We propose a sequential optimization framework that resolves global recovery via tractable local compensation. This approach effectively intercepts the error propagation chain at its source.
- **Efficiency & Versatility:** LaCo establishes a practical and scalable recovery paradigm, drastically minimizing resource requirements (memory & data) while offering flexible deployment options for diverse real-world scenarios.

2 Related works

2.1 Model Pruning

To mitigate the substantial memory and latency burdens of LLMs, model compression has evolved into a critical research area, encompassing techniques such as quantization (Lin et al., 2024), distillation (Gou et al., 2021), and pruning (Cheng et al., 2024). Among these, **pruning** is particularly pivotal as it directly reduces parameter counts to achieve tangible inference acceleration. Existing pruning methodologies are typically categorized by their granularity: *unstructured pruning* (e.g., Wanda (Sun et al., 2023)) removes individual weights for maximum flexibility; *structured pruning* (e.g., LLM-Pruner (Ma et al., 2023), FLAP (An et al., 2024)) targets coherent components like heads for hardware efficiency; and *semi-structured pruning* (e.g., SparseGPT (Frantar and Alistarh, 2023)) enforces N:M sparsity patterns. A standard pruning pipeline comprises two critical stages: *weight removal* and *performance recovery*. While the aforementioned methods have achieved significant strides in the removal stage (i.e., identifying unimportant parameters), they often leave the subsequent question of performance recovery under-addressed. Crucially, such removal-centric paradigms inevitably lead to disrupted internal representations, creating a gap that removal strategies alone cannot bridge. This establishes the essential motivation for our work: **effective post-pruning recovery is not an optional enhancement, but a fundamental requirement for successful deployment.**

2.2 Performance Compensation Strategies

To bridge this performance gap, existing research primarily diverges into two streams: *task-adaptive retraining* and *integrated reconstruction*. Yet, cur-

rent paradigms fail to offer a balanced solution that guarantees faithful, general, and efficient restoration. **Task-Adaptive Retraining methods**, exemplified by LoRA (Hu et al., 2022), operate by freezing the sparse backbone and optimizing auxiliary low-rank adapters on downstream data. While standard for adaptation, this mechanism constitutes an indirect compensation that bypasses rather than repairs the disrupted internal representations. Consequently, it inevitably introduces task bias—sacrificing the model's original generality—while incurring high labeled data costs that undermine the efficiency gains sought by pruning. **Integrated reconstruction methods** like FLAP (An et al., 2024) and SparseGPT (Frantar and Alistarh, 2023) incorporate compensation terms during the pruning phase. While efficient, these strategies are inherent to specific pruning metrics. This lack of generality prevents their application as a plug-and-play recovery module for other pruning paradigms. In contrast to these paradigms, we frame recovery as a hierarchical representation alignment problem. LaCo explicitly pursues fidelity, generality, and efficiency by sequentially aligning the pruned model's hidden states to those of the original dense model layer by layer.

3 Method

3.1 Problem Statement

We begin by formalizing the post-pruning compensation problem. Consider an L-layer model $\mathcal{M}$ with input x and output y. Each layer is represented by a transformation $\phi_k(\cdot;\theta_k)$ parameterized by weights θ_k. The forward propagation of the original model follows:

$$h_0 = x, \quad h_k = \phi_k(h_{k-1};\theta_k), \quad h_L = y. \quad (1)$$

After pruning, we obtain a sparse model $\mathcal{M}'$ with parameters θ'_k, whose forward propagation is given by:

$$h'_0 = x, \quad h'_k = \phi_k(h'_{k-1};\theta'_k), \quad h'_L = y'. \quad (2)$$

The primary goal of post-pruning recovery is to minimize the *output discrepancy* $\mathcal{D}$ between the dense and sparse models:

$$\min_{\theta'} \mathcal{D} = \|y - y'\| = \|h_L - h'_L\|. \quad (3)$$

To understand the mechanism of performance degradation, we decompose the layer-wise discrepancy via the triangle inequality:

$$\begin{aligned}\|h_k - h'_k\| &= \|\phi_k(h_{k-1};\theta_k) - \phi_k(h'_{k-1};\theta'_k)\| \\ &\le \underbrace{\|\phi_k(h_{k-1};\theta_k) - \phi_k(h'_{k-1};\theta_k)\|}_{\textbf{Input Error}} \\ &\quad + \underbrace{\|\phi_k(h'_{k-1};\theta_k) - \phi_k(h'_{k-1};\theta'_k)\|}_{\textbf{Local Error}}.\end{aligned} \quad (4)$$

This decomposition distinguishes two error sources: the *input error* stemming from the discrepancy in input hidden states, and the *local error* introduced directly by the weight change at the current layer.

Leveraging the Lipschitz continuity of Transformer layers (proof in Appendix A.1), we upper-bound the input error term with a Lipschitz constant K_k:

$$\|\phi_k(h_{k-1};\theta_k) - \phi_k(h'_{k-1};\theta_k)\| \le K_k \|h_{k-1} - h'_{k-1}\|. \quad (5)$$

For the local error, we define ϵ_k as the upper bound of the output discrepancy caused by pruning at layer k. Considering that layer inputs in LLMs are typically bounded due to normalization, we formulate this bound as:

$$\|\phi_k(h;\theta_k) - \phi_k(h;\theta'_k)\| \le \epsilon_k, \quad \forall h \in \mathcal{H}, \quad (6)$$

where $\mathcal{H}$ denotes the bounded domain of valid hidden states.

Substituting the Lipschitz bound (Eq. 5) and the local error definition (Eq. 6) into the decomposition (Eq. 4), we obtain the following recursive bound:

$$\|h_k - h'_k\| \le K_k \|h_{k-1} - h'_{k-1}\| + \epsilon_k. \quad (7)$$

By resolving this recursion from the first layer to the last, the final output discrepancy is bounded by:

$$\mathcal{D} = \|y - y'\| \le \sum_{l=1}^{L} \epsilon_l \left(\prod_{j=l+1}^{L} K_j \right). \quad (8)$$

Equation 8 reveals a crucial theoretical insight: the final output error stems from the layer-wise accumulation and progressive amplification of local error ϵ_l. Specifically, a local error ϵ_l introduced at an early layer l does not remain static; it is amplified by the product of Lipschitz constants $\prod_{j=l+1}^{L} K_j$ of all subsequent layers, potentially leading to catastrophic deviation at the final output.

Consequently, effective recovery requires a strategy that goes beyond global optimization. To minimize the total discrepancy $\mathcal{D}$, one must suppress the local error ϵ_l at each layer explicitly.

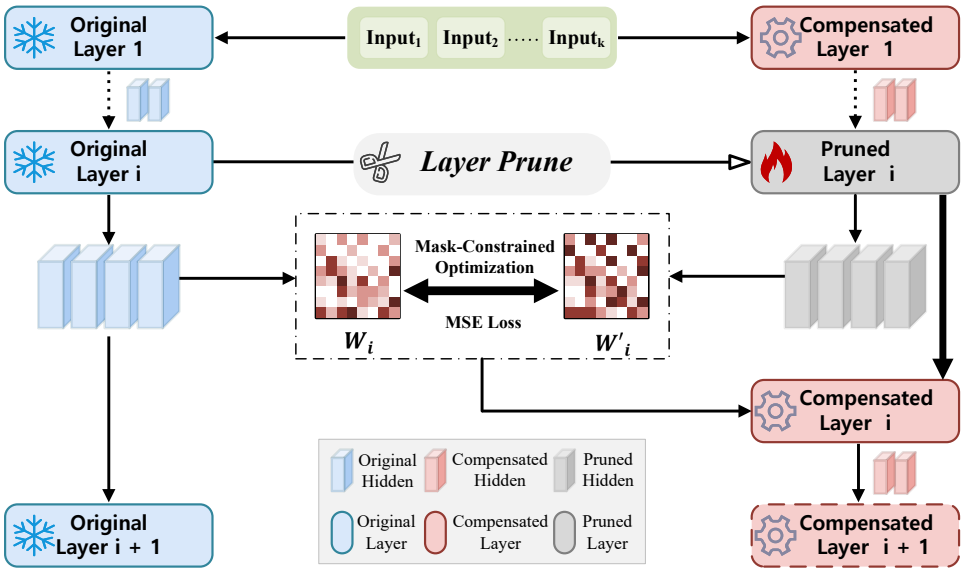


Figure 2: **Overview of the LaCo framework.** LaCo employs a sequential, layer-wise compensation strategy to restore the performance of pruned LLMs. For each target layer i, we formulate a **mask-constrained optimization** question that minimizes the Mean Squared Error (MSE) between the sparse hidden states and the original reference states. By optimizing only the remaining weights (W_i') while keeping the pruning mask fixed, LaCo effectively reconstructs the layer's functional behavior before propagating the compensated output to the subsequent layer.

3.2 Layer-wise Compensation Algorithm

Guided by the theoretical derivation in Eq. 8, minimizing the global discrepancy $\mathcal{D}$ necessitates the rigorous suppression of the local error ϵ_l at each layer.

However, directly optimizing all ϵ_l jointly is intractable due to the high-dimensional parameter space and complex inter-layer dependencies. Crucially, the recursive nature of the error bound suggests a pivotal strategy: decoupling the optimization. If the input to layer l is aligned with that of the dense model (i.e., $h'_{l-1} \approx h_{l-1}$), the global recovery problem collapses into a series of independent, tractable local regression subproblems.

To this end, we propose LaCo, which decomposes the global problem into sequential mask-constrained regression subproblems. This approach utilizes the compensated output of each layer to establish a faithful input baseline for the next, enabling tractable layer-wise optimization.

3.2.1 Single-Layer Compensation via Mask-Constrained Regression

We focus on the optimization of a specific layer l. Given the input activation $X \in \mathbb{R}^{B\times S\times d}$ derived from the preceding compensated layer (where $X \approx h_{l-1}$), we aim to align the sparse layer's representation with the original model. Specifically, we treat the dense output $Y = \phi_l(X;\theta_l)$ as the reference target and optimize the sparse parameters θ'_l to reconstruct output hidden state.

Since the pruning mask M_l is fixed, this is formulated as a mask-constrained regression problem:

$$\begin{aligned} \min_{\theta'_l} \quad & \mathcal{L}_{\text{local}}(\theta'_l) = \|Y - \phi_l(X;\theta'_l)\|_F^2 \\ \text{s.t.} \quad & \text{supp}(\theta'_l) \subseteq \text{supp}(M_l). \end{aligned} \tag{9}$$

By solving Eq. 9 via gradient descent, we leverage the expressive capacity of the remaining weights to minimize the layer-wise reconstruction error ϵ_l. This process produces a compensated output $h'_l = \phi_l(X;\theta'_l)$ that is functionally aligned with the original representations, serving as the rectified input for the subsequent layer.

3.2.2 Sequential Propagation for Global Recovery

To extend the local optimization to the entire model, LaCo employs a forward-chaining mechanism, as detailed in Algorithm 1. Instead of treating layers independently, the process iterates strictly from the first layer to the last. By utilizing the compensated output of the current layer (h'_l) as the input for the subsequent layer ($l+1$), we ensure that the optimization premise is continuously satisfied.

This design maintains a critical theoretical invariant throughout the model depth:

$$\|h'_{l-1} - h_{l-1}\| \approx 0 \quad \Rightarrow \quad \|h'_l - h_l\| \approx 0. \tag{10}$$

By preserving this condition step-by-step, LaCo effectively intercepts the error propagation path at its source, preventing local errors from accumulating into catastrophic global degradation.

Algorithm 1 Layer-wise Compensation (LaCo)

Require: Dense parameters $\{\theta_l\}$, Sparse parameters $\{\theta'_l\}$, Calibration Set $\mathcal{X}$, Batch size B, Learning rate η

Ensure: Compensated sparse parameters $\{\theta'_l\}$

Initialize hidden states $H_0 = H'_0 = \text{LoadBatch}(\mathcal{X})$
for $l = 1$ to L **do**
  ▷ *Phase 1: Target Generation*
  Compute dense target: $Y \leftarrow \phi_l(H_{l-1}; \theta_l)$
  Set current input: $X \leftarrow H'_{l-1}$
  ▷ *Phase 2: Mask-Constrained Optimization*
  for step $= 1$ to K **do**
    Compute Loss: $\mathcal{L} \leftarrow \|Y - \phi_l(X; \theta'_l)\|_F^2$
    Update: $\theta'_l \leftarrow \text{Optimizer}(\theta'_l, \nabla\mathcal{L}, \eta)$
    Constraint: $\theta'_l \leftarrow \theta'_l \odot M_l$
  end for
  ▷ *Phase 3: State Propagation*
  Update sparse state: $H'_l \leftarrow \phi_l(X; \theta'_l)$
  Update dense state: $H_l \leftarrow Y$
  Free memory for H_{l-1}, H'_{l-1}
end for
return $\{\theta'_l\}_{l=1}^{L}$

3.2.3 Algorithm Procedure and Practical Considerations

Algorithm 1 outlines LaCo's streaming execution, which caches only current-layer activations. This design limits peak memory to $\mathcal{O}(B \cdot S \cdot d_{\text{hidden}})$,where B is the batch size and S is the sequence length. Unlike global fine-tuning, LaCo's memory overhead is independent of model depth L, enabling drastic reductions in resource requirements

It is worth noting that LaCo is agnostic to the pruning stage. It functions effectively as a *post-pruning compensation* for already pruned models (where all masks M_l are pre-determined) or as an *on-the-fly enhancement* integrated into the pruning pipeline (where θ'_l is compensated immediately after being pruned). This flexibility allows LaCo to serve as a universal plug-in module for diverse pruning frameworks.

4 Experiments

4.1 Experimental settings

Pruning Paradigms. We validate LaCo across three distinct paradigms: (1) unstructured pruning using Wanda (Sun et al., 2023) at 20%, 50%, and 70% sparsity; (2) semi-structured pruning via SparseGPT (Frantar and Alistarh, 2023) with 2:4 and 4:8 patterns; (3) structured pruning employing LLM-Pruner (Ma et al., 2023) and FLAP (An et al., 2024) at 20%,50%,and 70% ratios.

Baselines. We benchmark our method primarily against LoRA (Hu et al., 2022), the standard for efficient fine-tuning, while also incorporating FLAP (An et al., 2024) as a specialized baseline for structured pruning scenarios.

Datasets and Evaluation Metrics. We assess restoration quality using both language modeling and reasoning capabilities. We report perplexity (PPL) on WikiText2 (Merity et al., 2016), PTB (Marcus et al., 1993), and C4 (Raffel et al., 2020). For downstream tasks, we evaluate zero-shot accuracy on seven benchmarks: BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2020), ARC-e (Clark et al., 2018), ARC-c, RTE (Dagan et al., 2005), and OpenBookQA (Mihaylov et al., 2018).

Implementation Details. We use the training split of WikiText2 for calibration. All experiments are conducted with a fixed random seed and full implementation details will be released upon acceptance.

4.2 Main Results

We evaluate LaCo across three pruning paradigms, with results summarized in Tables 1 and 2.

Structured Pruning (Most Challenging). Removing cohesive structural components (via LLM-Pruner and FLAP) typically triggers severe functional degradation. As observed in Table 1, at 70% sparsity, the uncompensated pruned model suffers a catastrophic collapse (PPL > 1000). While LoRA provides partial recovery, **LaCo** demonstrates superior restoration, significantly suppressing PPL and surpassing LoRA by **+2.72%** in mean accuracy. Even at 50% sparsity under the FLAP backbone, LaCo (47.43%) consistently outperforms both the internal bias-compensated baseline (42.47%) and LoRA (46.83%), proving its efficacy in rectifying severe representation shifts.

Unstructured and Semi-structured Robustness. LaCo exhibits remarkable versatility across granular pruning patterns. In **unstructured pruning** (Wanda), LaCo excels particularly in the high-sparsity regime (70%). While the pruned baseline's PPL spikes to 192.01, LaCo effectively intercepts error propagation, stabilizing PPL at **66.82** and outperforming LoRA by a clear margin (48.46% vs. 47.30%). Similarly, in **N:M semi-structured**

Table 1: Performance comparison on Structured Pruning paradigms. We evaluate LLM-Pruner (20%, 50%, 70%) and FLAP (50%, 70%). We explicitly compare the raw pruning (No Bias), the standard method (Base), and compensation strategies. **Bold** indicates the best result in each group. Gray rows indicate the backbone paradigm.

		Perplexity (PPL) ↓				Commonsense Reasoning Accuracy ↑								
Paradigm	**Method**	**Wiki2**	**PTB**	**C4**	**Mean**	**BoolQ**	**PIQA**	**RTE**	**Hella.**	**Wino.**	**OBQA**	**ARC-e**	**ARC-c**	**Mean**
Dense	LLaMA-2-7B	16.76	74.51	14.86	35.38	79.82	76.91	68.59	74.01	66.06	44.20	72.20	44.03	65.73
Pruning Backbone: LLM-Pruner														
20% Sparsity	LLM-Pruner (Base)	24.28	103.81	18.82	48.97	67.92	73.62	53.43	66.93	61.96	39.40	64.30	35.15	57.84
	LoRA	19.44	89.60	15.44	41.49	**74.98**	74.48	49.46	68.18	**62.75**	39.20	66.60	31.95	58.45
	LaCo (Ours)	**18.52**	**88.08**	**16.29**	**40.96**	71.25	**75.21**	**49.46**	**68.95**	61.25	**42.80**	**68.90**	**40.02**	**59.72**
50% Sparsity	LLM-Pruner (Base)	101.14	270.87	60.73	144.25	**57.52**	61.58	**55.23**	38.11	51.85	31.00	38.30	21.50	44.39
	LoRA	33.95	128.51	25.59	62.68	54.77	**71.48**	48.38	50.37	51.70	34.00	52.40	25.85	48.61
	LaCo (Ours)	**28.28**	**126.04**	**27.88**	**60.74**	54.28	69.12	53.07	**51.89**	**52.09**	**37.00**	**54.80**	**28.93**	**50.15**
70% Sparsity	LLM-Pruner (Base)	962.22	1699.69	439.69	1033.86	51.62	53.34	45.49	27.39	**49.88**	25.00	26.90	20.90	37.56
	LoRA	64.89	216.29	61.39	114.19	50.90	**61.47**	42.40	34.35	45.67	31.60	40.40	21.16	40.99
	LaCo (Ours)	**51.51**	**211.23**	**55.11**	**105.95**	**53.55**	61.35	**52.35**	**36.94**	49.72	**32.20**	**41.90**	**21.67**	**43.71**
Pruning Backbone: FLAP														
50% Sparsity	FLAP	114.22	446.63	112.50	224.45	48.23	59.91	**49.82**	42.70	52.88	29.60	40.70	24.40	43.53
	FLAP *	57.49	282.09	63.35	134.31	39.02	60.40	46.93	39.99	52.88	32.60	60.40	26.20	42.47
	LoRA	36.47	138.89	39.54	71.64	45.32	**65.86**	47.29	47.16	**53.99**	**36.20**	**65.86**	27.05	46.83
	LaCo (Ours)	**31.42**	**135.24**	**37.50**	**68.05**	**54.98**	63.67	47.29	**47.68**	53.59	33.80	63.67	**27.30**	**47.43**
70% Sparsity	FLAP	5769.67	9619.49	5928.47	7105.88	43.85	51.78	47.29	25.98	49.01	25.80	25.20	21.59	36.31
	FLAP *	1574.63	1877.09	1316.37	1589.36	37.83	52.09	47.29	24.93	50.20	26.80	25.20	**21.93**	35.78
	LoRA	633.36	851.76	508.47	664.53	40.55	54.39	51.62	**30.12**	49.17	28.80	33.50	20.73	38.61
	LaCo (Ours)	**79.02**	**341.90**	**105.58**	**175.50**	**43.55**	**56.43**	**53.43**	31.97	**50.51**	**30.60**	**37.90**	21.25	**40.71**

* *Indicates the baseline using FLAP's internal bias compensation mechanism.*

settings (SparseGPT), LaCo maintains the highest stability. Notably, in the 4:8 pattern, LaCo aligns the sparse weights to the original representations more effectively than LoRA, yielding a substantially lower perplexity (**40.00** vs. 41.63). Collectively, these results establish LaCo as a robust and universal paradigm for mitigating diverse pruning-induced degradations.

Generalizability across Architectures. To verify cross-architecture robustness, we extend our evaluations to **Qwen-2.5-7B** and the larger **LLaMA-2-13B** backbones. LaCo demonstrates exceptional robustness, particularly at extreme 70% sparsity. On Qwen-2.5, it slashes PPL from 158.42 (Wanda) to **43.09**, boosting zero-shot accuracy by **+11%**. Similarly, on LLaMA-2-13B, LaCo suppresses the baseline's catastrophic collapse (PPL 235.15 $\rightarrow$ **51.94**) and achieves a remarkable **13.9%** accuracy gain. These results confirm LaCo's efficacy in preserving capabilities across diverse backbones and scales.

4.3 Analysis of Layer-wise Compensation

Layer-wise Performance Analysis. To validate the efficacy of our layer-wise compensation mechanism, we conducted a cumulative restoration experiment: for each step k (ranging from 1 to 32), we compensated only the first k layers and evaluated global perplexity. As shown in Fig. 3, the perplexity exhibits a sharp decline as the compensation depth increases. Moreover, this recovery trajectory remains highly consistent across diverse datasets,

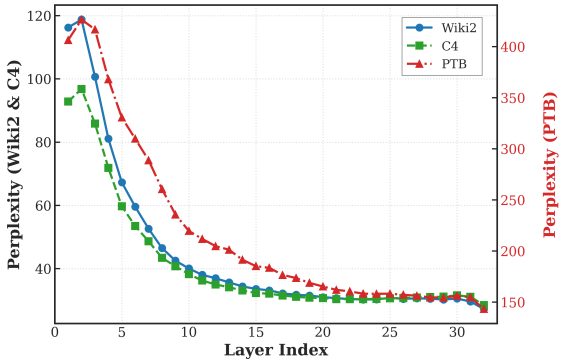


Figure 3: **Layer-wise Reconstruction Analysis.** The PPL restoration trajectory across 32 layers demonstrates that LaCo effectively mitigates errors, particularly in the early layers.

demonstrating the universality and robustness of our approach. These results confirm that LaCo effectively mitigates error propagation at its onset, preventing severe deviations from accumulating in the early layers. This finding highlights that prioritizing early-layer alignment is essential for efficient model recovery.

Layer-wise Fidelity Analysis. We further delve into the representational dynamics of LaCo by examining structural integrity across network depth. Our analysis reveals that LaCo achieves **sparsity-agnostic stability**, maintaining a near-perfect Centered Kernel Alignment (CKA $>$ 0.98) even at extreme sparsity levels. In contrast, monolithic fine-tuning baselines often fail to preserve structural fidelity, especially in deeper layers where errors have

Table 2: Performance comparison on Unstructured (Wanda) and Semi-structured (SparseGPT) pruning paradigms. We evaluate Wanda at 20%, 50%, and 70% sparsity, and SparseGPT at 2:4 and 4:8 patterns. **Bold** indicates the best result in each group. Gray rows indicate the backbone paradigm.

		Perplexity (PPL) ↓				Commonsense Reasoning Accuracy ↑								
Paradigm	**Method**	**Wiki2**	**PTB**	**C4**	**Mean**	**BoolQ**	**PIQA**	**RTE**	**Hella.**	**Wino.**	**OBQA**	**ARC-e**	**ARC-c**	**Mean**
Dense	LLaMA-2-7B	16.76	74.51	14.86	35.38	79.82	76.91	68.59	74.01	66.06	44.20	72.20	44.03	65.73
Pruning Backbone: Wanda (Unstructured)														
20% Sparsity	Wanda (Base)	18.04	81.43	15.84	38.44	78.72	**76.63**	71.12	**73.91**	64.88	44.80	70.90	43.35	65.54
	LoRA	16.59	**69.88**	**10.94**	**32.47**	76.24	76.60	69.45	72.74	64.56	41.80	**71.30**	41.89	64.32
	LaCo (Ours)	**15.82**	74.16	14.95	34.98	**79.79**	76.39	**72.92**	73.87	**65.59**	**45.00**	**71.30**	**44.03**	**66.11**
50% Sparsity	Wanda (Base)	21.72	102.92	19.06	47.90	76.12	**74.69**	59.93	**68.83**	62.59	39.00	67.50	37.71	60.79
	LoRA	17.59	77.88	15.33	36.93	74.34	74.18	55.23	68.50	**62.83**	**40.80**	66.70	36.52	59.88
	LaCo (Ours)	**17.37**	**77.72**	**15.18**	**36.76**	**76.57**	74.32	**62.09**	68.77	62.43	39.80	**67.50**	**39.25**	**61.34**
70% Sparsity	Wanda (Base)	114.76	376.86	84.41	192.01	59.45	56.78	52.71	30.21	51.38	24.20	33.40	17.83	40.74
	LoRA	33.27	183.78	23.35	80.13	63.09	61.40	52.71	41.98	50.62	**33.80**	**50.90**	23.89	47.30
	LaCo (Ours)	**27.26**	**144.69**	**28.51**	**66.82**	**64.13**	**64.23**	**56.68**	**44.66**	**51.78**	32.20	50.00	**23.98**	**48.46**
Pruning Backbone: SparseGPT (Semi-structured)														
2:4 Sparsity	SparseGPT (Base)	29.10	122.41	23.47	58.33	**72.08**	71.00	62.09	60.66	61.88	35.40	61.00	34.30	57.30
	LoRA	20.87	90.60	**18.37**	43.28	70.43	72.39	68.59	64.05	60.88	35.60	62.30	30.46	58.09
	LaCo (Ours)	**19.15**	**89.95**	18.60	**42.57**	71.83	**72.68**	**69.68**	**64.42**	**60.93**	**39.00**	**67.60**	**37.03**	**60.39**
4:8 Sparsity	SparseGPT (Base)	24.65	106.31	20.45	50.47	**74.98**	**73.11**	70.40	66.07	**62.90**	**40.20**	63.50	34.73	**60.73**
	LoRA	21.10	89.63	**14.16**	41.63	73.18	69.13	**71.12**	66.51	61.72	30.00	63.70	33.36	58.59
	LaCo (Ours)	**18.32**	**86.27**	15.39	**40.00**	68.56	70.24	70.45	**67.48**	59.12	36.80	**67.20**	**36.62**	59.56

already cascaded. A comprehensive breakdown of the Mean Squared Error (MSE) trajectories and CKA comparisons is provided in Appendix B.

Module-wise Sensitivity Analysis. To further disentangle the source of pruning-induced degradation within a single transformer block, we investigate the relative sensitivity of different architectural modules. Specifically, we conduct an ablation study by applying LaCo exclusively to the Multi-Head Attention (MHA) modules (while freezing the pruned MLP weights) and vice versa. As reported in Table 4, recovering only the Attention modules yields a significantly lower perplexity compared to recovering only the MLP modules across all evaluated datasets. This observation suggests that self-attention mechanisms are inherently more vulnerable to sparsity-induced structural distortions than feed-forward networks. Nevertheless, while attention recovery dominates the performance gain, it remains insufficient to fully bridge the degradation gap. The substantial improvement achieved by the joint recovery of both modules underscores our core premise: pruning errors compound not only across layers but also across intra-layer modules, making comprehensive compensation strictly necessary.

4.4 Hyperparameter sensitivity

We systematically investigate the sensitivity of our method to three critical hyperparameters: the calibration size, the learning rate, and the training epochs. Additionally, we analyze the convergence behavior to identify the optimal configuration that balances performance and efficiency.

Calibration Size. We assess the impact of calibration dataset size (ranging from 128 to 8,192) on restoration quality. As illustrated in Fig. 4 (Left), performance improves consistently with sample size before saturating. A critical comparison reveals that while LoRA typically requires ∼50,000 training samples for effective fine-tuning, LaCo achieves comparable performance with fewer than 1,400 samples across varied settings. To balance computational overhead with stability, we adopt 2,048 samples as the default setting. This implies that our method achieves a ∼**25**× data efficiency gain compared to standard LoRA fine-tuning.

Learning Rates. We further investigate the sensitivity of our method to different learning rates. As illustrated in Fig. 4, we evaluate the model performance across a range of learning rates. The results indicate that the performance peaks at 5×10^{-5}, while deviating from this value leads to suboptimal convergence. Therefore, we set the learning rate to 5×10^{-5} for all subsequent experiments.

Training Cost. We analyze the training convergence by monitoring model performance from 0 to 15 epochs. As shown in Fig. 4, the model exhibits a rapid performance recovery during the initial 5 epochs. Subsequently, the convergence rate slows down, and the performance plateaus after the 10th epoch, remaining stable up to epoch 15. To achieve the best trade-off between training efficiency and

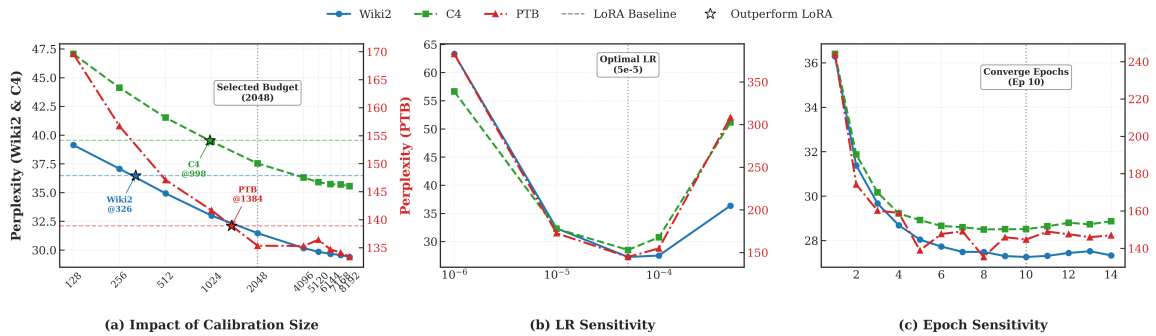


Figure 4: **Sensitivity Analysis.** (Left) **Impact of Calibration Size**: Performance comparison against LoRA across varying sample sizes, showing an efficiency balance at 2048 samples. (Middle) **Learning Rate Sensitivity**: Identification of the optimal learning rate (5e-5) for stable convergence. (Right) **Epoch Sensitivity**: Training convergence trajectory demonstrating stability around epoch 10.

Table 3: Condensed performance comparison. We report the **Mean Perplexity** (↓) and **Mean Accuracy** (↑) across all datasets/tasks. The detailed breakdown is available in the Appendix C

	Qwen-2.5-7B		Llama-2-13B	
Method	**Mean PPL ↓**	**Mean Acc ↑**	**Mean PPL ↓**	**Mean Acc ↑**
Dense Model	23.18	64.25	34.24	68.43
Unstructured Pruning (Wanda)				
20% Sparsity				
Base	23.33	**64.10**	34.37	69.08
+ LoRA	**21.94**	63.81	**34.64**	69.02
+ **LaCo**	23.27	64.08	34.90	**70.24**
50% Sparsity				
Base	28.59	62.16	41.21	65.94
+ LoRA	27.32	62.39	41.13	65.00
+ **LaCo**	**26.38**	**63.02**	**38.24**	**66.28**
70% Sparsity				
Base	158.42	43.87	235.15	43.88
+ LoRA	48.67	53.47	71.86	54.27
+ **LaCo**	**43.09**	**55.10**	**51.94**	**57.77**
Semi-structured Pruning (SparseGPT)				
2:4 Pattern				
Base	37.24	58.19	63.79	55.60
+ LoRA	35.56	58.54	59.96	56.90
+ **LaCo**	**33.29**	**59.19**	**58.10**	**58.16**
4:8 Pattern				
Base	31.31	60.50	49.13	60.30
+ LoRA	30.95	59.98	50.13	**60.98**
+ **LaCo**	**29.94**	**60.98**	**49.09**	60.88

model performance, we select 10 epochs as the optimal setting.

4.5 Efficiency Analysis

We conduct a comprehensive efficiency profiling on LLaMA-2-7B pruned by Wanda at 70% sparsity. As detailed in Table 5, LaCo is compared against LoRA across three critical dimensions: data, memory, and time.

Extreme Data Efficiency (∼25× Less Data). LaCo exhibits exceptional sample efficiency. While LoRA typically requires ∼50k supervised samples for convergence to fully recover capabilities, LaCo outperforms its performance with only 2,048 unlabeled samples—a ∼25× reduction in data dependency (Table 5, #Samples). This makes LaCo uniquely viable for practical deployment in specialized domains (e.g., medical or legal models) where high-quality calibration data is strictly scarce, expensive, or restricted by privacy concerns.

Recovered Module	**WikiText2 ↓**	**PTB ↓**	**C4 ↓**
MLP Only	43.19	421.75	58.04
Attention Only	38.36	280.12	35.97
Both (LaCo)	**27.25**	**144.69**	**28.50**

Table 4: Ablation study on intra-layer module recovery. We report the perplexity when LaCo is applied exclusively to MLP, Attention, or both modules. Joint recovery yields the best performance, indicating error compounding across modules.

Democratizing LLM Recovery (∼7× Less Memory). The most transformative advantage is hardware democratization. By decoupling optimization into layer-wise problems, LaCo slashes peak memory usage to 6.2 GB, a ∼7× reduction from LoRA's 43.7 GB (Table 5, Peak Memory). Consequently, while LoRA demands data-center GPUs (e.g., A100), LaCo enables restoring 7B models on consumer-grade hardware (e.g., RTX 4090), drastically lowering the entry barrier for model compression.

Highly Favorable Time-for-Space Trade-off. LaCo's sequential design increases total training time (6h 50m vs. 4h 20m). We argue this is an excellent trade-off: the modest time cost unlocks the ability to recover large models under extreme memory constraints where global fine-tuning would be impossible, making LaCo uniquely practical for resource-limited settings.

Table 5: Comparison of training efficiency between LaCo and LoRA. We report the calibration data size, peak GPU memory usage, and total training time. Memory usage is measured on a single NVIDIA A100 GPU.

Method	#Samples	Peak Memory ↓	Training Time	Memory Saving ↑
LoRA	49,760	43.7 GB	**4h 20m**	1.0×
LaCo (Ours)	**2,048**	**6.2 GB**	6h 50m	**7.0**×

5 Conclusion

This work pinpoints the layer-wise accumulation and amplification of local errors as the fundamental cause of performance collapse in pruned LLMs. To address this, we introduce LaCo, a recovery framework that shifts the paradigm from global fine-tuning to hierarchical representation alignment. Our key finding is that the functional integrity of pruned LLMs can be effectively restored through layer-wise compensation. This challenges the prevailing assumption that meaningful recovery necessitates expensive end-to-end retraining. By sequentially aligning each layer's hidden states with the original dense model, LaCo directly intercepts the error propagation chain, preserving the model's internal structural fidelity. Extensive experiments demonstrate that LaCo consistently outperforms state-of-the-art baselines across diverse pruning paradigms, especially under high sparsity. Crucially, it achieves this with dramatically lower resource demands: reducing peak memory usage by $\sim$7$\times$ and data requirements by $\sim$25$\times$ compared to standard fine-tuning, thereby making high-quality recovery feasible on consumer-grade hardware. LaCo thus provides a practical and efficient pathway toward democratizing compressed LLM deployment. Future work will focus on extending the layer-wise alignment principle to other compression techniques, such as quantization.

6 Limitations

Our work has several limitations. First, while LaCo significantly recovers performance under high sparsity (e.g., 70%), a gap to the dense model remains (Tables 1–3), suggesting a recovery ceiling inherent to severe weight removal. Precisely quantifying this performance ceiling is crucial for developing more efficient model compression methods. Second, our focus is on pruning recovery; its interaction with other prevalent compression techniques like quantization remains unexplored. Third, validation is conducted on Transformer-based LLMs; its efficacy on emerging architectures (e.g., SSMs) or at extreme depth requires further study.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant Nos. 62225205, 62472162, and 62532005).

References

Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. 2024. Fluctuation-based adaptive structured pruning for large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 10865–10873.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Hao Chen, Haoze Li, Zhiqing Xiao, Lirong Gao, Qi Zhang, Xiaomeng Hu, Ningtao Wang, Xing Fu, and Junbo Zhao. 2025a. Alps: Attention localization and pruning strategy for efficient adaptation of large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 11764–11780.

Xiaodong Chen, Yuxuan Hu, Xiaokang Zhang, Yanling Wang, Cuiping Li, Hong Chen, and Jing Zhang. 2025b. P^2 law: Scaling law for post-training after model pruning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5668–5686, Vienna, Austria. Association for Computational Linguistics.

Hongrong Cheng, Miao Zhang, and Javen Qinfeng Shi. 2024. A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

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

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine learning challenges workshop*, pages 177–190. Springer.

Xuan Ding, Rui Sun, Yunjian Zhang, Xiu Yan, Yueqi Zhou, Kaihao Huang, Suzhong Fu, Angelica I. Aviles-Rivero, Chuanlong Xie, and Yao Zhu. 2025. A Sliding Layer Merging Method for Efficient Depth-Wise Pruning in LLMs. *Preprint*, arXiv:2502.19159.

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.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International journal of computer vision*, 129(6):1789–1819.

Jialong Guo, Xinghao Chen, Yehui Tang, and Yunhe Wang. 2025. SlimLLM: Accurate Structured Pruning for Large Language Models. *Preprint*, arXiv:2505.22689.

Soufiane Hayou, Eugenio Clerico, Bobby He, George Deligiannidis, Arnaud Doucet, and Judith Rousseau. 2021. Stable resnet. In *International Conference on Artificial Intelligence and Statistics*, pages 1324–1332. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Hyunjik Kim, George Papamakarios, and Andriy Mnih. 2021. The lipschitz constant of self-attention. In *International Conference on Machine Learning*, pages 5562–5571. PMLR.

Hanyu Lai, Xiao Liu, Junjie Gao, Jiale Cheng, Zehan Qi, Yifan Xu, Shuntian Yao, Dan Zhang, Jinhua Du, Zhenyu Hou, Xin Lv, Minlie Huang, Yuxiao Dong, and Jie Tang. 2025. A survey of post-training scaling in large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2771–2791, Vienna, Austria. Association for Computational Linguistics.

Chi-Heng Lin, Shangqian Gao, James Seale Smith, Abhishek Patel, Shikhar Tuli, Yilin Shen, Hongxia Jin, and Yen-Chang Hsu. 2025. MoDeGPT: Modular Decomposition for Large Language Model Compression. *Preprint*, arXiv:2408.09632.

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

Deyuan Liu, Zhanyue Qin, Hairu Wang, Zhao Yang, Zecheng Wang, Fangying Rong, Qingbin Liu, Yanchao Hao, Xi Chen, Cunhang Fan, Zhao Lv, Zhiying Tu, Dianhui Chu, Bo Li, and Dianbo Sui. Pruning via Merging: Compressing LLMs via Manifold Alignment Based Layer Merging. *Preprint*, arXiv:2406.16330.

Songtao Liu and Peng Liu. 2025. High-Layer Attention Pruning with Rescaling. *Preprint*, arXiv:2507.01900.

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

Mitch Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

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

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. GPT-4 Technical Report. *Preprint*, arXiv:2303.08774.

Xianbiao Qi, Jianan Wang, Yihao Chen, Yukai Shi, and Lei Zhang. 2023. Lipsformer: Introducing lipschitz continuity to vision transformers. *arXiv preprint arXiv:2304.09856*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

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

Dmitriy Shopkhoev, Ammar Ali, Magauiya Zhussip, Valentin Malykh, Stamatios Lefkimmiatis, Nikos Komodakis, and Sergey Zagoruyko. 2025. ReplaceMe: Network Simplification via Layer Pruning and Linear Transformations. *Preprint*, arXiv:2505.02819.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2023. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, and 1 others. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Pingjie Wang, Ziqing Fan, Shengchao Hu, Zhe Chen, Yanfeng Wang, and Yu Wang. 2025. Reconstruct the Pruned Model Without Retraining. *IEEE Journal of Selected Topics in Signal Processing*, pages 1–12.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. In *Advances in Neural Information Processing Systems*, volume 33, pages 5776–5788. Curran Associates, Inc.

Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. 2019. Understanding and improving layer normalization. *Advances in neural information processing systems*, 32.

Qwen: An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Hao Yu and Jianxin Wu. 2023. Compressing Transformers: Features Are Low-Rank, but Weights Are Not! *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(9):11007–11015.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Mingyang Zhang, Hao Chen, Chunhua Shen, Zhen Yang, Linlin Ou, Xinyi Yu, and Bohan Zhuang. 2024. Loraprune: Structured pruning meets low-rank parameter-efficient fine-tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3013–3026.

Longguang Zhong, Fanqi Wan, Ruijun Chen, Xiaojun Quan, and Liangzhi Li. 2025. BlockPruner: Fine-grained pruning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5065–5080, Vienna, Austria. Association for Computational Linguistics.

A Theoretical Analysis of Lipschitz Continuity

In this section, we provide a comprehensive theoretical justification for the error propagation analysis presented in Sec. 3.1. We first prove that the Transformer layer transformation $\phi_l(\cdot)$ satisfies the Lipschitz continuity condition (existence), and then discuss why its Lipschitz constant K_l is bounded and typically close to 1 in well-trained models (magnitude).

A.1 Proof of Existence

We begin by formally defining the property and verifying it for Transformer components.

Definition and Properties. A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is K-Lipschitz continuous if there exists a constant $K \geq 0$ such that for all $x_1, x_2 \in \mathbb{R}^n$:

$$\|f(x_1) - f(x_2)\| \leq K\|x_1 - x_2\|, \tag{11}$$

where $\|\cdot\|$ denotes the Euclidean norm. The infimum of such K is denoted as $\text{Lip}(f)$. Key algebraic properties for neural networks include:

1. **Composition**: $\text{Lip}(g \circ f) \leq \text{Lip}(g) \cdot \text{Lip}(f)$.
2. **Addition**: $\text{Lip}(f + g) \leq \text{Lip}(f) + \text{Lip}(g)$.

Component-wise Analysis. A Transformer layer ϕ_l comprises Multi-Head Attention (MHA), Feed-Forward Networks (FFN), LayerNorm (LN), and residual connections.

- **Linear Layers:** For $f(x) = Wx + b$, $\text{Lip}(f) = \|W\|_2$ (spectral norm). Since weights in pre-trained LLMs are bounded, $\|W\|_2$ is finite.
- **Activation Functions:** Standard activations (ReLU, GeLU, SiLU) have bounded derivatives (e.g., $\text{Lip}(\text{ReLU}) = 1$, GeLU bounded by ≈ 1.1). Thus, they are Lipschitz continuous.
- **Self-Attention & Normalization:** LayerNorm is Lipschitz on bounded inputs. The MHA mechanism, composed of linear projections and Softmax, is Lipschitz continuous provided inputs are bounded, as rigorously derived in (Kim et al., 2021; Qi et al., 2023).
- **Residual Connections:** For a residual block $y = x + F(x)$, we have $\text{Lip}(y) \leq 1 + \text{Lip}(F)$ by the addition property.

Since ϕ_l is a composition of these atomic operations, it satisfies the Lipschitz property with a constant $K_l = \text{Lip}(\phi_l)$. This justifies the inequality:

$$\|\phi_l(h_{l-1}) - \phi_l(h'_{l-1})\| \leq K_l\|h_{l-1} - h'_{l-1}\|. \tag{12}$$

A.2 Discussion on the Magnitude of K_l

Having established the existence of K_l, a critical question remains: *Is K_l sufficiently small to prevent the error bound from diverging exponentially with depth?* We argue that in well-trained LLMs, K_l remains well-bounded, typically close to 1, due to the following structural constraints.

Constraint from Layer Normalization. Theoretical studies (Ba et al., 2016; Xu et al., 2019) suggest that Layer Normalization acts as a constraint mechanism. By re-centering and re-scaling activations, LN effectively restricts the spectral norm of the layer's Jacobian, ensuring that the local expansion factor K_l does not grow arbitrarily.

Impact of Residual Connections. Deep networks rely on residual connections $h_l = h_{l-1} + F_l(h_{l-1})$ for trainability. It is well-observed that the residual branch F_l tends to learn small fluctuations around the identity mapping to facilitate gradient flow (Hayou et al., 2021). Consequently, $\text{Lip}(\phi_l) \approx 1 + \epsilon$, where ϵ is small. This design encourages signal isometry, implying stable rather than explosive error propagation.

Empirical Evidence. The successful convergence of original LLMs (e.g., LLaMA) serves as strong empirical evidence that their weight matrices are well-conditioned. If K_l were significantly larger than 1, gradients would explode during pre-training. Since we inherit these stable weights, the assumption of a manageable K_l holds valid.

Summary. In conclusion, the Lipschitz continuity of Transformer layers is structurally guaranteed, and the magnitude of the Lipschitz constant is effectively bounded by normalization and residual designs. This confirms that the error accumulation described in Eq. 5 remains linear or sub-linear, validating the feasibility of our layer-wise compensation strategy.

B Detailed Analysis of Representational Fidelity

To investigate the internal recovery mechanisms of *LaCo* compared to baselines, we conduct a layer-wise analysis using two complementary metrics: Mean Squared Error (MSE) and Centered Kernel Alignment (CKA).

Rationale for Metric Selection.

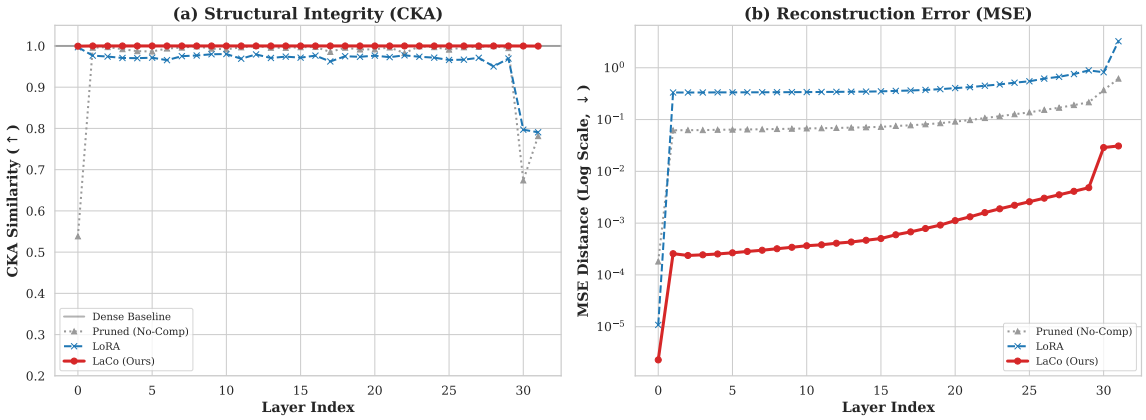
- **MSE (Numerical Precision):** We utilize MSE to quantify the element-wise reconstruction loss of hidden states. This metric is essential for tracing the *magnitude of error propagation*, revealing how local numerical deviations accumulate and amplify as they pass through the network layers.
- **CKA (Structural Integrity):** Relying solely on MSE can be misleading, as low numerical error does not guarantee semantic preservation. Therefore, we employ CKA to measure the similarity between representational geometries, independent of isotropic scaling or rotation. High CKA scores indicate that the model preserves the original *feature manifold* and co-activation patterns ("thinking logic"), which is critical for complex reasoning tasks.

Based on these metrics, we analyze the representational quality of *LaCo*, *LoRA*, and the *Pruned* baseline from three distinct perspectives, as visualized in Figure 5:

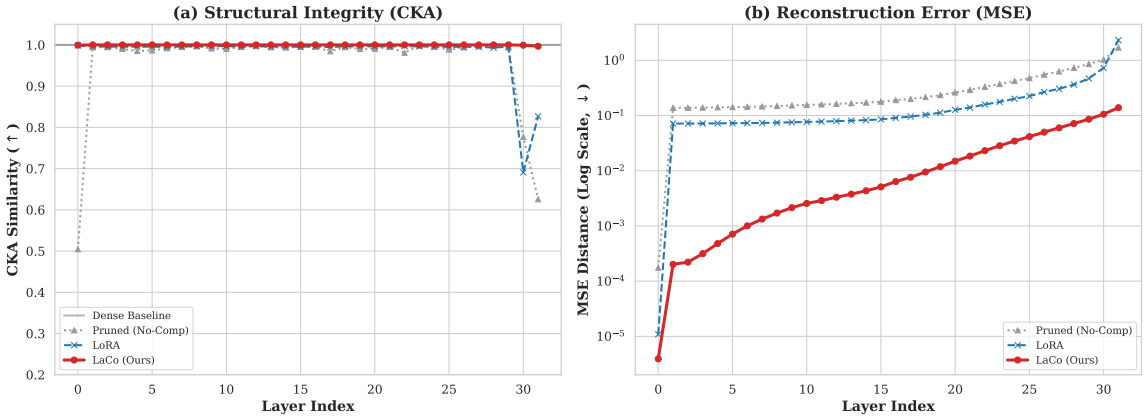
1. Sparsity-Agnostic Structural Stability. As pruning rates increase, the structural integrity of the uncompensated model collapses (CKA < 0.3 at 70% sparsity). While LoRA also suffers from degradation at high sparsity levels ($CKA \approx 0.89$), **LaCo** demonstrates remarkable stability, maintaining a CKA score near 1.0 consistently across 20%, 50%, and 70% sparsity. This suggests our method effectively decouples structural preservation from pruning intensity.

2. Mechanism of Error Interruption. In terms of MSE, LaCo consistently outperforms LoRA at lower sparsity levels. Under the extreme 70% regime, although LoRA exhibits lower MSE in intermediate layers due to global relaxation, LaCo achieves superior convergence in the final layers. This indicates that LaCo prioritizes structural alignment over intermediate numerical fitting, ultimately minimizing the cumulative error at the network's output.

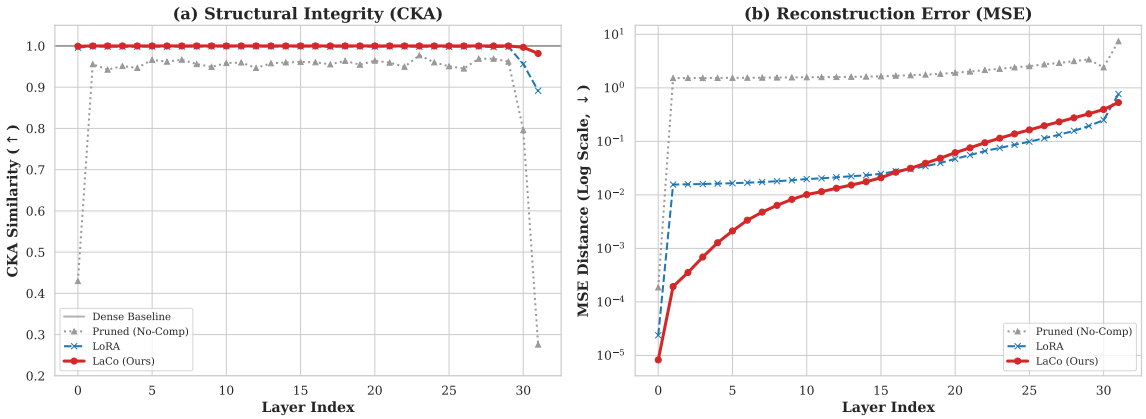
3. Robustness in Deep Layers. A critical observation is the "tail collapse" in the baseline models. Both the pruned model and LoRA show a distinct drop in representational fidelity in the final transformer blocks (Layers 28–31). LaCo, conversely, maintains robustness throughout the deep network. By ensuring high-quality feature representations at the final layer, LaCo provides a superior input to the Language Model Head, directly contributing to the improved perplexity scores reported in the main results.



(a) **20% Sparsity**: At low compression, both methods maintain high fidelity, though LaCo leads in MSE convergence.



(b) **50% Sparsity**: Divergence begins. LaCo maintains structural integrity ($CKA \approx 1.0$), while LoRA shows fluctuations.



(c) **70% Sparsity**: Critical regime. Pruned baseline collapses; LoRA degrades in deep layers; LaCo remains robust.

Figure 5: **Layer-wise Representational Fidelity Across Sparsity Regimes.** Comparison of Structural Integrity (CKA, Left) and Reconstruction Error (MSE, Right) at 20%, 50%, and 70% sparsity. **(a)** At low sparsity, performance is comparable. **(b)** As sparsity increases, LoRA (blue dashed) begins to exhibit structural drift. **(c)** At extreme sparsity, the uncompensated model (grey) collapses. LaCo (red solid) acts as an effective error interrupter, preserving deep-layer robustness where global fine-tuning fails.

C More results of Qwen-2.5-7B and LLaMA-2-13B

Table 6: Performance comparison of Qwen-2.5-7B on Unstructured (Wanda) and Semi-structured (SparseGPT) pruning paradigms. We evaluate Wanda at 20%, 50%, and 70% sparsity, and SparseGPT at 2:4 and 4:8 patterns. **Bold** indicates the best result in each group. Gray rows indicate the backbone paradigm.

		Perplexity (PPL) ↓				Commonsense Reasoning Accuracy ↑								
Paradigm	**Method**	**Wiki2**	**PTB**	**C4**	**Mean**	**RTE**	**BoolQ**	**Hella**	**Wino**	**OBQA**	**PIQA**	**ARC-e**	**ARC-c**	**Mean**
Dense	Qwen-2.5-7B	17.23	34.08	18.21	23.18	88.09	85.87	78.52	57.06	46.20	79.24	43.64	35.41	64.25
Pruning Backbone: Wanda (Unstructured)														
20% Sparsity	Wanda (Base)	17.44	34.24	18.31	23.33	88.09	85.20	78.06	56.51	46.40	79.11	43.69	**35.75**	64.10
	LoRA	**16.04**	35.86	**17.92**	**23.27**	87.36	**87.83**	77.44	**56.99**	46.00	**79.13**	**43.98**	31.73	63.81
	LaCo (Ours)	17.29	**34.08**	18.44	**23.27**	**88.81**	85.26	**78.13**	56.67	**47.20**	78.99	42.26	35.32	**64.08**
50% Sparsity	Wanda (Base)	20.85	42.55	22.36	28.59	83.75	84.04	70.16	56.35	46.20	75.96	**43.81**	**37.03**	62.16
	LoRA	19.32	40.22	22.42	27.32	84.48	**85.69**	**74.03**	**57.30**	44.40	**77.30**	41.96	33.96	62.39
	LaCo (Ours)	**18.87**	**39.37**	**20.90**	**26.38**	**86.28**	84.40	73.53	56.12	**46.80**	76.97	43.64	36.43	**63.02**
70% Sparsity	Wanda (Base)	124.11	227.07	124.09	158.42	52.71	62.60	33.20	50.91	26.40	59.98	36.03	29.10	43.87
	LoRA	35.40	71.92	38.70	48.67	73.98	71.97	**52.28**	**56.91**	38.00	**69.01**	37.60	28.03	53.47
	LaCo (Ours)	**28.18**	**66.19**	**34.89**	**43.09**	**81.23**	**77.25**	50.70	54.54	**38.60**	67.93	**40.45**	**30.12**	**55.10**
Pruning Backbone: SparseGPT (Semi-structured)														
2:4 Sparsity	SparseGPT (Base)	25.00	55.47	31.26	37.24	83.03	**83.12**	58.37	55.09	40.60	70.81	41.54	**32.94**	58.19
	LoRA	26.74	52.09	27.85	35.56	**83.20**	81.65	62.08	**57.56**	39.80	71.85	**41.99**	30.15	58.54
	LaCo (Ours)	**22.01**	**51.11**	**26.76**	**33.29**	81.23	82.60	**63.04**	57.46	**44.20**	**72.53**	41.75	30.72	**59.19**
4:8 Sparsity	SparseGPT (Base)	22.14	45.87	25.92	31.31	**84.48**	82.78	63.65	56.04	**44.40**	73.34	**43.60**	**35.67**	60.50
	LoRA	23.39	**44.92**	24.53	30.95	82.31	83.92	66.07	56.75	42.60	**75.78**	40.57	31.83	59.98
	LaCo (Ours)	**20.35**	45.33	**24.13**	**29.94**	83.75	**84.83**	**66.81**	**57.30**	43.60	74.45	42.85	34.22	**60.98**

Table 7: Performance comparison of Llama-2-13B on Unstructured (Wanda) and Semi-structured (SparseGPT) pruning paradigms. We evaluate Wanda at 20%, 50%, and 70% sparsity, and SparseGPT at 2:4 and 4:8 patterns. **Bold** indicates the best result in each group. Gray rows indicate the backbone paradigm.

		Perplexity (PPL) ↓				Commonsense Reasoning Accuracy ↑								
Paradigm	**Method**	**Wiki2**	**PTB**	**C4**	**Mean**	**RTE**	**BoolQ**	**Hella**	**Wino**	**OBQA**	**PIQA**	**ARC-e**	**ARC-c**	**Mean**
Dense	Llama-2-13B	10.98	82.58	9.17	34.24	76.17	80.09	77.30	70.48	44.20	79.33	74.58	45.31	68.43
Pruning Backbone: Wanda (Unstructured)														
20% Sparsity	Wanda (Base)	11.17	**82.68**	**9.27**	34.37	80.51	80.64	74.56	71.27	**45.00**	79.50	75.04	46.08	69.08
	LoRA	11.19	83.39	9.35	34.64	82.67	78.42	72.38	**72.30**	43.60	**79.94**	**76.56**	**46.32**	69.02
	LaCo (Ours)	**10.99**	84.34	9.37	**34.90**	**83.26**	**82.15**	**75.33**	72.09	43.60	77.98	77.85	49.69	**70.24**
50% Sparsity	Wanda (Base)	13.16	99.26	11.20	41.21	76.53	**78.46**	71.86	66.32	41.00	**77.46**	73.06	**42.83**	65.94
	LoRA	13.06	99.42	**10.90**	41.13	**80.87**	74.87	67.95	65.03	**42.80**	76.50	70.45	41.50	65.00
	LaCo (Ours)	**12.08**	**91.54**	11.09	**38.24**	78.43	77.03	**70.47**	**69.43**	41.80	75.74	**74.88**	42.42	**66.28**
70% Sparsity	Wanda (Base)	60.12	580.28	65.06	235.15	52.71	62.02	32.73	53.20	28.80	60.29	41.25	20.05	43.88
	LoRA	20.91	177.61	**17.06**	71.86	60.98	61.03	53.16	59.27	36.80	66.12	**62.54**	34.22	54.27
	LaCo (Ours)	**16.60**	**121.09**	18.13	**51.94**	**62.71**	**63.06**	**62.82**	**63.35**	**39.40**	**72.56**	62.27	**35.98**	**57.77**
Pruning Backbone: SparseGPT (Semi-structured)														
2:4 Sparsity	SparseGPT (Base)	**16.99**	157.20	17.18	63.79	53.43	67.19	57.20	**63.93**	**39.80**	70.22	62.25	30.80	55.60
	LoRA	18.07	145.10	16.70	59.96	**65.81**	66.36	58.48	61.85	38.60	**71.16**	62.13	30.84	56.90
	LaCo (Ours)	17.09	**143.06**	**14.15**	**58.10**	64.98	**69.69**	**61.32**	61.80	38.60	70.07	**62.29**	**36.55**	**58.16**
4:8 Sparsity	SparseGPT (Base)	**14.46**	119.21	**13.71**	49.13	60.29	75.57	65.76	**65.98**	37.80	73.52	67.59	35.92	60.30
	LoRA	14.99	121.10	14.30	50.13	**68.70**	71.92	62.57	67.96	**41.80**	**73.81**	**72.31**	**36.83**	**60.98**
	LaCo (Ours)	14.57	**118.94**	13.75	**49.09**	61.40	**76.16**	**65.49**	65.51	40.60	73.59	68.81	35.49	60.88