LLM-Barber: A One-Shot Block-Aware Rebuilder for Sparsity Mask in Large Language Models

***Abstract*—Large language models (LLMs) have seen substantial growth, necessitating efficient model pruning techniques. Existing post-training pruning methods primarily measure weight importance in converged dense models, often overlooking changes in weight significance during the pruning process, leading to performance degradation. To address this issue, we present LLM-Barber (A One-Shot Block-Aware Rebuilder for Sparsity Mask), a simple yet effective pruning approach that rebuilds the sparsity mask of pruned models without any retraining or weight reconstruction. LLM-Barber incorporates block-aware error optimization across Self-Attention and MLP blocks, facilitating global performance optimization. We are the first to employ the product of weights and gradients as a pruning metric in the context of LLM post-training pruning. This enables accurate identification of weight importance in massive models and significantly reduces computational complexity compared to methods using second-order information. Our approach significantly reduces memory usage and computational demands, making it more efficient for deployment on accelerators and edge devices. Experiments on LLaMA3-8B with 50% sparsity show up to a 19x improvement in performance when incorporating LLM-Barber into the pruning process, while achieving a state-of-the-art perplexity of 9.45 on WikiText-2 and an average zero-shot performance of 55.21% across various tasks in post-training pruning domain.**

***Index Terms*—Large Language Models, Post-training Pruning, Sparsity Mask Rebuilding, Low-resource Deployment.**

I. Introduction

LLMs have become foundational in artificial intelligence due to their impressive performance on various tasks. However, the increasing size and complexity of models, such as GPT-175B [1] with 175 billion parameters, pose significant challenges related to extensive computational and storage demands. Consequently, efficient model compression strategies are crucial for enabling the practical edge deployment.

Network pruning shrinks network sizes by setting specific weights to zero, effectively removing them from the network. However, current pruning methods face two major challenges. First, traditional layer-aware pruning methods focus on individual layers, neglecting inter-layer dependencies, which increases error accumulation (blue arrows in Figure 1(a)). Second, as shown in Figure 1(b), conventional methods typically build the sparsity mask only once, ignoring the changes of weight significance in post-pruning stage. This oversight can lead to improper identification of salient weights and subsequent performance degradation.

To address these limitations, we propose LLM-Barber, a novel and straightforward approach to rebuild sparsity mask of pruned networks without requiring any retraining or weight reconstruction. Firstly, LLM-Barber integrates pruning across Self-Attention and MLP blocks. This approach mitigates error

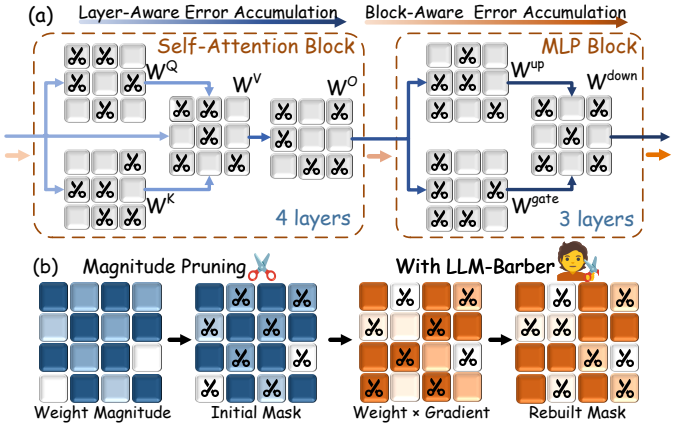

Fig. 1: The benefits of integrating LLM-Barber into the pruning process: (a) Transition from the layer-aware to block-aware error accumulation for an optimized solution. (b) Rebuilding sparsity mask using a novel pruning metric.

accumulation, as evidenced by the lighter orange arrows, promoting more global optimization. Secondly, LLM-Barber identifies weights that, although initially non-salient without a sparsity mask, gain significance in post-pruning. As shown in Figure 1(b), varying color shades represent the relative importance scores of different weights. LLM-Barber accurately rebuilds masks for high-score weights (deeply shaded) while pruning newly identified low-score weights (lightly shaded). Thirdly, LLM-Barber employs the product of weights and gradients as pruning metric. While this metric has been applied in other contexts [2], [3], we are the first to leverage it in the context of post-training compression for LLMs, reducing computational complexity compared to second-order methods [4]. Finally, LLM-Barber efficiently rebuilds sparsity masks in a one-shot manner, surpassing fine-tuning methods in efficiency while maintaining high accuracy. This efficiency makes it more favorable for accelerating LLMs on hardware platforms. To sum up, the key contributions are fourfold:

- **Block-Aware Global Optimization:** We are among the first to introduce a block-aware reconstruction problem that integrates sparsity across the Self-Attention and MLP blocks, achieving global optimization in pruning.
- **Rebuilding Sensitive Regions:** By identifying and retaining significant weights post-pruning while dynamically adjusting sparsity masks, we enhance model performance through optimal sparsity reallocation.
- **Innovative Pruning Metric:** We are the first to apply the product of weights and gradients as a pruning metric

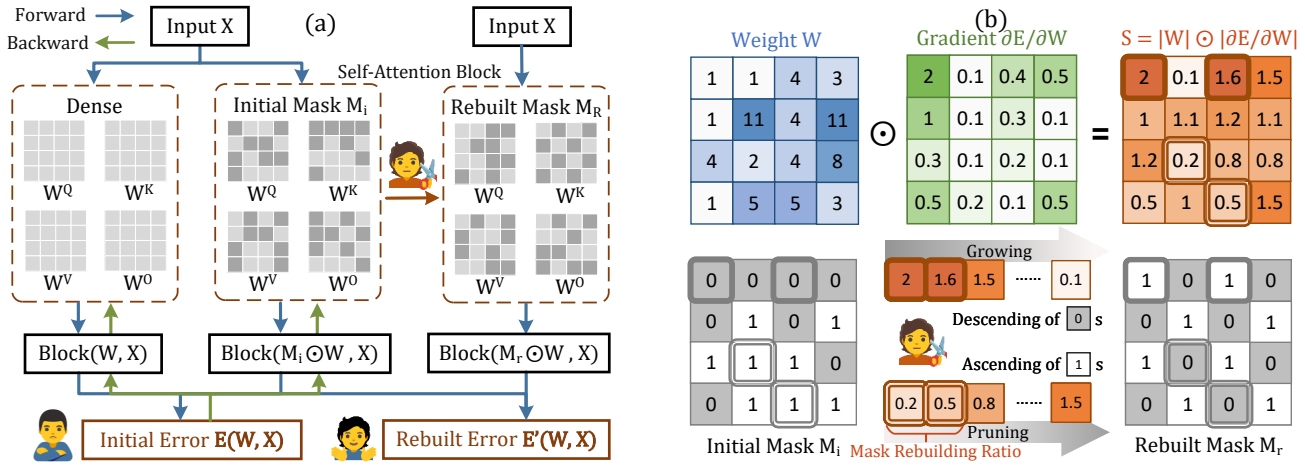

Fig. 2: The workflow of LLM-Barber. (a) illustrates the process of block-aware reconstruction error and gradient calculation for each linear weight. (b) shows pruning metric computation and sparsity mask rebuilding.

in post-training compression, leveraging first-order Taylor series for importance evaluation to reduce computational complexity compared to Hessian-based approaches.

- **Efficiency for Hardware Acceleration:** LLM-Barber's one-shot pruning method is more efficient than fine-tuning approaches, making it highly suitable for accelerating LLMs on hardware platforms. It achieves state-of-the-art performance in perplexity and zero-shot tasks, setting new benchmarks in LLM post-training pruning.

II. Related Work

A. LLMs Pruning and Sparsity

Network pruning reduces deep neural networks by eliminating unnecessary weights, with methods categorized into parameter-efficient fine-tuning (PEFT) and post-training approaches. PEFT begins with an initialized sparse network and refines it through iterative processes [5]. LoRA [6] adapts pre-trained models to specific tasks by injecting trainable rank decomposition matrices. Dynamic Spase No Training [7] prunes and grows weights to minimize reconstruction error. However, fine-tuning often requires ample data and can degrade performance. Post-training approach removes weights from a pre-trained model. SparseGPT [4] uses Hessian-based metrics and subsequent residual weight updates, Wanda [8] introduces metric using weight-activation products.

B. Model Compression Strategy

Compression is key to reducing the memory and computational demands of model. Layer-aware strategies, originating from Optimal Brain Damage [9] and Optimal Brain Surgeon [10], have been enhanced by recent works like GPTQ [11] using second-order information. However, block-aware strategies typically yield better accuracy recovery. For instance, APTQ [12] applies global quantization to attention mechanisms, BESA [13] uses block-wise sparsity allocation. Our method leverages block-aware pruning to optimize global performance across blocks, effectively balancing efficiency and accuracy.

III. LLM-Barber

A. Preliminaries

LLM pruning removes weights from dense networks while minimizing output discrepancies. However, this process is computationally intensive for large-scale models, as it requires addressing the layer-aware reconstruction problem [10]. This section will reanalyzes the layer-aware reconstruction error and the application of Taylor expansion at dense networks.

Layer-aware Reconstruction Error. For linear projection layer weight $\mathbf{W}$ of shape $(C_{\text{out}}, C_{\text{in}})$, where C_{out}, C_{in} indicates the output and input channels. With N calibration samples and sequence length L, the input activation is denoted as $\mathbf{X}$ with the shape of $(C_{\text{in}}, N \times L)$. Layer-aware reconstruction error $\mathbf{E}$ is defined as the ℓ_2 norm of output difference between dense and sparse layers:

$$\mathbf{E} = ||\mathbf{W}\mathbf{X} - \widehat{\mathbf{W}}\mathbf{X}||_2^2, \tag{1}$$

where $\widehat{\mathbf{W}}$ is the element-wise product of $\mathbf{W}$ and a binary sparsity mask $\mathbf{M}(i,j) \in \{0,1\}$ of shape $(C_{\text{out}}, C_{\text{in}})$, in context with mask selection and without weight reconstruction:

$$\mathbf{E} = ||\mathbf{W}\mathbf{X} - (\mathbf{W} \odot \mathbf{M})\mathbf{X}||_2^2. \tag{2}$$

The goal is to find an optimal sparsity mask $\mathbf{M}$ that reduces model complexity while preserving predictive accuracy.

Taylor Expansion at Dense Networks. For a dense network $\widehat{\mathbf{W}}_{\text{dense}}$ at a local minimum, the reconstruction error $\mathbf{E}$ can be expanded into its Taylor series with respect to sparsity mask $\mathbf{M}$, ignoring terms beyond second order:

$$\mathbf{E} = \mathbf{E_d} + \frac{\partial \mathbf{E}}{\partial \mathbf{W}}\Delta\mathbf{W} + \frac{1}{2}\Delta\mathbf{W}^T\mathbf{H}\Delta\mathbf{W}. \tag{3}$$

Without a sparsity mask in dense model, we can simply assign all-one matrix to the mask M, thereby yielding the zeroth-order terms $\mathbf{E_d} = 0$. The first-order derivative $\partial\mathbf{E}/\partial\mathbf{W}$ vanishes when training converged [4], [10], leaving only the computationally expensive second-order terms involving a large Hessian matrix which are challenging for layer-wise reconstruction and channel-wise independence assumption.

TABLE I: WikiText-2 perplexity comparison at 50% sparsity rate. WR represents weight reconstruction.

	LLaMA1		LLaMA2		LLaMA3	OPT	
Method	**7B**	**13B**	**7B**	**13B**	**8B**	**6.7B**	**13B**
Dense	5.677	5.091	5.472	4.884	6.136	10.86	10.13
Magnitude	17.26	20.14	16.03	6.827	205.5	9.7e2	1.2e4
w/ **LLM-Barber**	**7.332**	**6.089**	**7.170**	**5.955**	**10.98**	**13.12**	**15.52**
SparseGPT	7.201	6.194	7.005	6.036	9.399	11.59	11.15
SparseGPT *w/o* WR	7.545	6.311	7.413	6.134	9.994	13.13	15.76
w/ **LLM-Barber**	**7.159**	**6.125**	**7.004**	**5.929**	**9.348**	**11.95**	**11.93**
Wanda	7.254	6.152	6.920	5.972	9.821	11.98	11.93
w/ **LLM-Barber**	**7.118**	**6.091**	**6.868**	**5.918**	**9.451**	**11.95**	**11.71**

Bold show improvements with LLM-Barber, underscored indicate best performance.

B. A One-Shot Block-Aware Rebuilder for Sparsity Mask

In this work, we depart from existing post-training pruning methods in three key aspects: Firstly, we adopt a block-aware reconstruction error and apply a divide-and-conquer strategy to mitigate errors and computational costs. Secondly, we address the limitations of mask selection of pruning in dense networks due to the changeable significance of weights, by re-evaluating weight importance score in sparse networks and rebuilding the sparsity mask through targeted growth of salient weights and pruning of non-salient weights. Thirdly, our analysis reveals that with the advancement of LLMs, mask selection becomes increasingly critical in weight reconstruction. Thus, we prioritize the rebuilding of the sparsity mask and strip the reconstruction of weights. Base on these insights, we propose **LLM-Barber**, a **B**lock-**A**ware **Reb**uild**er** for Sparsity Mask in One-Shot without any fine-tuning or retraining.

Block-Aware Reconstruction Error. Building on the definitions in Eq. (1) and Eq. (2), we define the block-aware reconstruction error for a Self-Attention or MLP block:

$$\mathbf{E} = ||\text{Block}(\mathbf{W}, \mathbf{X}) - \text{Block}(\mathbf{W} \odot \mathbf{M}, \mathbf{X})||_2^2. \tag{4}$$

Evaluating reconstruction error across blocks, denoted as $\text{Block}(\cdot)$, allows us to achieve a globally optimal solution in Self-Attention and MLP blocks rather than layer-wise.

Taylor Expansion at Sparse Networks. Migrating Eq. (3) at sparse networks $\widehat{\mathbf{W}}_{\text{sparse}}$ with an initialization sparsity mask $\mathbf{M_i}$, we can obtain the Taylor series expansion as:

$$\mathbf{E} = \mathbf{E_s} + \frac{\partial \mathbf{E}}{\partial \mathbf{W}} \Delta \mathbf{W} + \frac{1}{2} \Delta \mathbf{W}^T \mathbf{H} \Delta \mathbf{W}. \tag{5}$$

The zeroth-order term $\mathbf{E_s}$ at sparse networks represents the reconstruction error after mask initialization:

$$\mathbf{E_s} = ||\text{Block}(\mathbf{W}, \mathbf{X}) - \text{Block}(\mathbf{W} \odot \mathbf{M_i}, \mathbf{X})||_2^2. \tag{6}$$

Assuming non-negligible zeroth-order terms, the first-order gradient in sparse networks remains significant even after convergence and can be efficiently accessed via PyTorch's Autograd. First-order information provides computational efficiency and operates independently of any reconstruction error. Therefore, second-order terms can be omitted when significant gradients are present, leading to the following change in reconstruction error due to mask rebuilding:

$$\Delta \mathbf{E} = (\partial \mathbf{E} / \partial \mathbf{W}) \cdot \Delta \mathbf{W}, \tag{7}$$

which delineates the importance score of weights during sparsity mask rebuilding.

Pruning Metric. For a Self-Attention or MLP Block with weights $\mathbf{W}$, first-order information suffices for block-aware reconstruction error in sparse networks. The change in weight magnitude during sparsity mask adjustment matches the weight's original magnitude ($|\Delta \mathbf{W}ij| = |\mathbf{W}ij|$). We thus assess the impact of mask rebuilding on reconstruction error by computing the product of the weight's magnitude and its gradient. The **importance score** for $\mathbf{W}_{ij}$ is:

$$\mathbf{S}_{ij} = |\mathbf{W}_{ij}| \cdot |(\partial \mathbf{E} / \partial \mathbf{W}_{ij})|, \tag{8}$$

where $|\cdot|$ represents the absolute value operator, and $\mathbf{E}$ denotes the block-aware reconstruction error. This metric prioritizes weights with both magnitudes and significant gradients.

Pruning Granularity. Choosing the right pruning granularity is crucial [8]. Traditional methods operate on a layer-wise [14], input-wise [4], or output-wise [8] basis, which mainly address the layer-aware reconstruction problem, known for its output channel independence. Wanda's output-wise ranking yields superior results compared to other methods. However, in LLM-Barber's block-aware framework, output channel independence is no longer applicable. Thus, LLM-Barber extends consideration to block-wise granularity, prioritizing all linear layers within a block. Our analysis of the four distinct granularity levels shows that optimal granularity depends on the specific sparse mask initialization, with detailed results discussed in the Ablation Study IV-D.

Mask Rebuilding. With the block-aware reconstruction error, gradient information, sparsity matrix, and granularity established, we proceed to rebuild the sparsity mask for each layer. Consider a cluster of weights $\mathbf{W^c}$ under a specific sparsity granularity and its corresponding sparsity mask $\mathbf{M^c}$ and pruning metric $\mathbf{S^c}$. We define the **growing criterion** and the **pruning criterion**:

$$gi, gj = \text{argmax } \mathbf{S^c}, \text{ if } \mathbf{M^c}_{gi,gj} = 0, \tag{9}$$

$$pi, pj = \text{argmin } \mathbf{S^c}, \text{ if } \mathbf{M^c}_{pi,pj} = 1. \tag{10}$$

The growing and pruning weights form a **mask rebuilding pair**, representing the interchange within the sparsity mask. The value of each pair is defined as the difference between the importance scores of the growing and pruning weights.

Algorithm 1 Pseudocode of LLM-Barber.

Input: Calibration samples $\mathbf{X}$, a block's weights $\{\mathbf{W}^l\}_{l=1}^L$ and initial masks $\{\mathbf{M_i}^l\}_{l=1}^L$, mask rebuilding ratio α.
Output: Rebuilt sparsity masks $\{\mathbf{M_r}^l\}_{l=1}^L$.

$\mathbf{E_i} \leftarrow \text{Block}(\mathbf{W}, \mathbf{X}) - \text{Block}(\mathbf{W} \odot \mathbf{M_i}, \mathbf{X})$
$\{\mathbf{G}^l\}_{l=1}^L \leftarrow$ BP for gradients via $\mathbf{E_i}$.
for l in $\{1, 2, \ldots, L\}$ **do**
  $\mathbf{M_r}^l \leftarrow \mathbf{M_i}^l$
  $\mathbf{S}^l \leftarrow |\mathbf{W}^l| \cdot |\mathbf{G}^l|$
  $N \leftarrow \mathbf{S}^l, \alpha$ via Eq.(11)
  for n in $\{1, 2, \ldots, N\}$ **do**
    Obtain index gi, gj, pi, pj via Eq.(9) (10)
    $\mathbf{M_r}^l_{pi,pj} = 1, \mathbf{M_r}^l_{gi,gj} = 0$
  end for
end for
$\mathbf{E_r} \leftarrow \text{Block}(\mathbf{W}, \mathbf{X}) - \text{Block}(\mathbf{W} \odot \mathbf{M_r}, \mathbf{X})$
Identify improvement by comparing $\mathbf{E_i}$, $\mathbf{E_r}$.
return rebuilt sparsity masks $\{\mathbf{M_r}^l\}_{l=1}^L$.

Our experiments show that LLM-Barber identifies varying proportions of salient weights based on the sparse mask's initialization method. To regulate mask rebuilding, we introduce a hyperparameter α, known as the mask rebuilding ratio. The number of mask rebuilding pairs N is calculated as:

$$N = \{\ i \mid \mathbf{S}_i^{\text{grow}} - \mathbf{S}_i^{\text{prune}} > 0\} \cdot \alpha, \tag{11}$$

where $\mathbf{S}^{\text{grow}} - \mathbf{S}^{\text{prune}}$ represents the value of the mask rebuilding pairs, where $\mathbf{S}^{\text{grow}}$ is arranged in descending order, and $\mathbf{S}^{\text{prune}}$ in ascending order. The subscript i denotes the number of values that exceeds zero, indicating that the growing weight is more important than the pruning weight.

C. Procedure

LLM-Barber is executed within a single global LLM forward pass, with local backward passes for gradient computation in each block. Figure 2 illustrates the LLM-Barber workflow, which consists of four stages:

(1) Sparsity Mask Initialization. Initialize a preliminary sparsity mask from the dense network by a post-training pruning technique. **(2) Block-aware Reconstruction Error Computation.** We use a block-aware reconstruction error to evaluate the discrepancy between the dense and sparse model outputs. **(3) Back-propagation for Gradients.** Gradients are automatically derived via back-propagation, and the product of weights and gradients serves as the pruning metric. **(4) Sparsity Mask Rebuilding.** Masks are sorted by pruning metric, unpruned weights in ascending order while pruned weights in descending order. We rebuild the weight masks by growing newly significant weights and pruning those became non-salient with a specific mask rebuilding ratio.

D. Structured N:M Sparsity

While LLM-Barber primarily targets unstructured sparsity, it can be adapted for structured N:M sparsity. Groups of M weights are pruned to retain only N non-zero weights. During mask rebuilding, it divides each M-group into N pairs (one pruned and one non-pruned weight), then sorts them by output channel to identify mask rebuilding pairs. This method optimizes sparsity mask, leveraging N:M sparsity while maintaining model performance.

IV. Experiment

A. Experiment Settings

Setup. LLM-Barber is implemented in Pytorch and utilized public model checkpoints from the HuggingFace library on a single 80GB NVIDIA A100 GPU. After mask initilization, LLM-Barber uniformly rebuilds sparsity masks in sequence, performing in one-shot without any fine-tuning.

Models & Datasets. LLM-Barber is evaluated on the LLaMA and OPT model family, including LLaMA-7B/13B [15], LLaMA2-7B/13B [16], LLaMA3-8B [17] as well as OPT-6.7B/13B [18]. Notably, LLM-Barber is broadly applicable to any Transformer-based LLMs with Self-Attention and MLP blocks. Following previous works, we use 128 segments of 2048 tokens from the C4 dataset [19] for mask rebuilding.

Evaluation. To comprehensively assess LLM-Barber, we conduct rigorous evaluations on *perplexity* and *zero-shot performance*. Perplexity is measured on the validation sets of benchmarks such as WikiText-2 [20], PTB [21], and C4 [19]. Zero-shot performance is assessed using EleutherAI LM Harness [22] across six tasks: BoolQ [23], RTE [24], HellaSwag [25], ARC Easy and Challenge [26], and OpenbookQA [27].

Baselines. The results of LLM-Barber are compared with the following established post-training pruning methods: (1) Magnitude Pruning [14] eliminates weights based only on their magnitudes; (2) SparseGPT [4] identifies weights importance by using second-order information; (3) Wanda [8] determines weights to be pruned by the weight multiplied by activation.

B. Main Results

Unstructured Sparsity. LLM-Barber effectively prunes the LLaMA and OPT models, achieving 50% unstructured sparsity without requiring supplementary weight reconstruction, as detailed in Table I. LLM-Barber demonstrates the capability to rebuild the sparsity masks initialized by other pruning methods in a single forward pass, significantly outperforming conventional pruning baselines. In the LLaMA3-8B model, LLM-Barber creates a new sparsity mask that reduces perplexity to 9.451, a substantial improvement over the Wanda baseline of 9.821. Notably, LLM-Barber achieves robust improvements even with poorly performing initial sparsity masks, such as magnitude pruning, where it reduces perplexity from 205.5 to 10.98, which is an impressive and substantial enhancement.

Zero-shot Performance. Following previous works [4], [8], we evaluated the LLaMA models on six diverse zero-shot tasks. The results are summarized in Table II, where models are pruned to unstructured 50% sparsity. Averaging the accuracy across the six evaluated tasks, it becomes apparent that LLM-Barber possesses the capability to identify a more effective network than those obtained via the initialization methods.

TABLE II: Zero-shot performance comparison of LLaMA series on six tasks at 50% sparsity rate.

Model	Method	BoolQ	RTE	HellaSwag	ARC-e	ARC-c	OBQA	Mean
LLaMA-7B	Dense	75.08	66.79	56.96	75.29	41.89	34.40	58.40
	Magnitude	54.61	54.51	45.47	58.75	33.45	22.60	44.89
	w/ **LLM-Barber**	**71.47**	**59.93**	**50.63**	**69.11**	**35.41**	**28.20**	**52.46**
	Wanda	70.98	55.23	51.90	69.44	36.86	28.60	52.17
	*w/***LLM-Barber**	**73.12**	**56.68**	51.80	**70.32**	**36.95**	**28.80**	**52.95**
LLaMA2-13B	Dense	80.58	65.34	60.05	79.38	48.46	35.20	61.50
	Magnitude	70.53	55.96	54.42	57.68	38.40	27.80	50.79
	w/ **LLM-Barber**	**81.10**	**60.29**	**55.67**	**75.34**	**40.28**	**31.20**	**57.31**
	Wanda	80.95	60.28	56.98	76.30	42.26	31.20	57.99
	w/ **LLM-Barber**	**80.98**	**62.82**	55.99	**76.39**	42.13	**31.40**	**58.29**
LLaMA3-8B	Dense	81.35	69.68	60.19	80.09	50.60	34.80	62.79
	Magnitude	42.87	53.07	29.85	46.59	25.09	22.00	36.57
	w/ **LLM-Barber**	**72.72**	**54.87**	**51.00**	**72.05**	**37.46**	**27.40**	**52.58**
	Wanda	78.41	60.29	51.20	71.38	40.10	29.40	55.13
	*w/***LLM-Barber**	**78.59**	**61.37**	51.01	**71.46**	39.85	29.00	**55.21**

Bold show improvements with LLM-Barber, underscored indicate best performance.

For tasks such as BoolQ, RTE, and ARC-e, LLM-Barber consistently outperforms the baseline pruning techniques across the entire LLaMA model suite. However, it is worth noting that there is no single universally superior performer for the remaining tasks in the evaluation set, with the initial pruning methods sometimes matching or even marginally exceeding the results obtained through LLM-Barber.

TABLE III: WikiText-2 perplexity performance for pruning LLaMA3-8B at varying sparsity rate.

Sparsity	60%	70%	80%	90%
Magnitude	3.39e4	1.62e6	8.55e7	2.26e7
w/ **LLM-Barber**	**28.14**	**2.08e2**	**1.09e3**	**7.55e4**
Wanda	23.57	1.28e2	8.54e2	1.28e4
w/ **LLM-Barber**	**22.04**	**1.07e2**	**5.70e2**	**7.07e3**

Varying Sparsity Levels. We conduct experiments on varying sparsity levels for unstructured pruning in LLaMA3-8B as shown in Table III. LLM-Barber consistently increases perplexity across all initialization methods, with magnitude pruning showing the most significant improvement.

TABLE IV: WikiText-2 perplexity comparison for pruning LLaMA family with structured N:M pattern.

Method	Sparsity	V1-7B	V1-13B	V2-7B
Magnitude	4:8	16.83	13.72	15.91
w/ **LLM-Barber**	4:8	**8.852**	**7.137**	**9.345**
SparseGPT	4:8	8.608	7.437	8.495
w/ **LLM-Barber**	4:8	**8.191**	**7.085**	**8.003**
Magnitude	2:4	42.56	18.32	37.76
w/ **LLM-Barber**	2:4	**11.04**	**9.006**	**13.47**
SparseGPT	2:4	11.55	9.116	10.94
w/ **LLM-Barber**	2:4	**10.14**	**8.517**	**9.806**

Structured N:M Sparsity. In contrast to unstructured sparsity, employing N:M fine-grained sparsity can provide more tangible acceleration benefits when leveraging NVIDIA Ampere's sparse tensor cores [28]. Therefore, we also evaluate the effectiveness of our LLM-Barber in partial LLaMA models on the N:M fine-grained sparsity pattern as shown in Table IV.

C. Mask Rebuilding Ratio Selection

A critical aspect of the LLM-Barber method lies in determining the optimal mask rebuilding ratio for peak accuracy. This is effectively achieved by analyzing the distribution of value magnitudes within mask rebuilding pairs, reflecting differences between growing and pruning importance scores. Notably, this distribution often exhibits distinct outliers, enabling rapid identification of suitable mask rebuilding ratios.

We plotted the score distribution of mask rebuilding pairs across various pruning granularities and initialization methods on LLaMA-7B, along with perplexity performance at various rebuilding ratios (Figure 3). The results reveal a strong correlation between the distribution of outliers in mask rebuilding pairs and the optimal mask rebuilding ratio. For instance, outliers are significantly distributed within the top 10% with Magnitude pruning, thus selecting a 10% mask rebuilding ratio yields the overall optimal solution. Similarly, the Wanda initialization method shows a notable outlier distribution around the 1% mark corresponds to optimal results near a 1% mask rebuilding ratio. Thus, LLM-Barber can preemptively narrow the search range for the optimal mask rebuilding ratio by analyzing outlier distributions, allowing for flexible adaptation to various reconstruction masks without extensive searching.

D. Ablation Study

Given the significant potential of LLM-Barber, we analyzed three critical factors to assess the robustness and effectiveness of our method: pruning granularity, pruning metric, and calibration data size. These factors were chosen to gain a deeper understanding of how different configurations affect the performance of pruned models and to demonstrate LLM-Barber's versatility across various settings.

Pruning Granularity. LLM-Barber dynamically selects different pruning granularities to adapt to varying initialization methods. In this paper, we evaluated the impact of

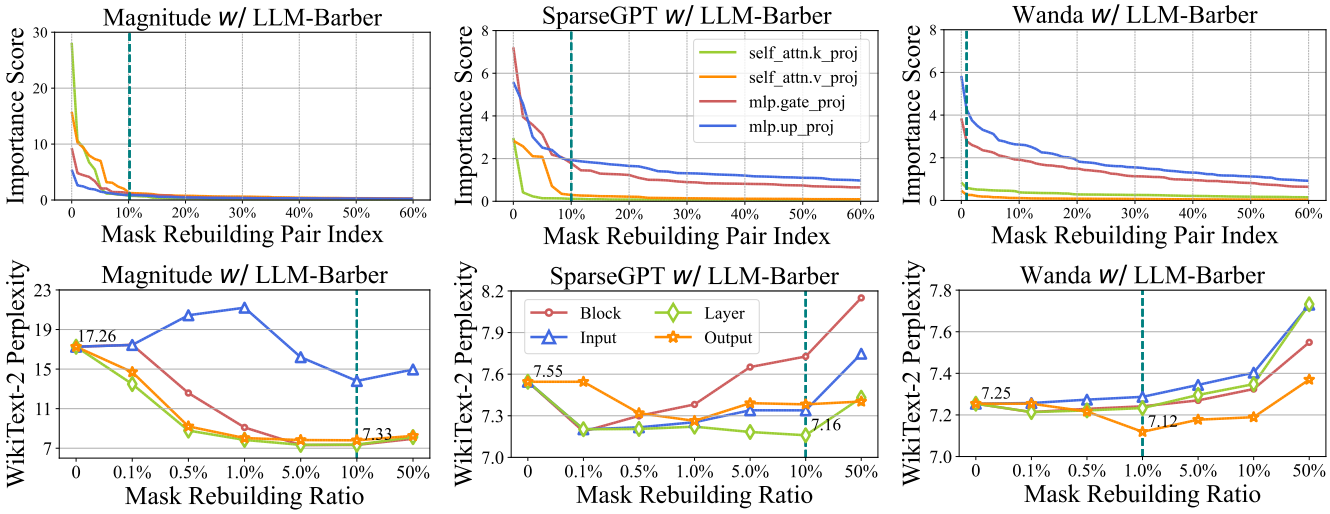

Fig. 3: The importance score distribution of mask rebuilding pairs and WikiText-2 perplexity results at varying pruning granularities in LLaMA-7B, with the green dashed line marking the optimal mask rebuilding ratio.

four levels of granularities: block-wise, layer-wise, input-wise, and output-wise pruning, as shown in Table V. For magnitude pruning, the block-wise granularity yields the best performance, while the output-wise granularity delivers the lowest perplexity for Wanda pruning. By tailoring the pruning granularity to the particular pruning approach, LLM-Barber can consistently achieve optimal model performance.

TABLE V: Pruning granularity ablation in LLaMA3-8B.

Method	Pruning Granularity			
w/ LLM-Barber	Block	Layer	Input	Output
Magnitude	**10.98**	11.06	72.48	11.81
SparseGPT	9.380	**9.348**	9.418	9.567
Wanda	9.626	9.633	9.849	**9.451**

Bold results show best granularity of each row.

Pruning Metric. We analyze the effect of different pruning metrics, weight magnitude, gradient magnitude, and product of weight and gradient. As shown in Table VI, the product of weight and gradient consistently outperforms the others, achieving the lowest perplexity of 10.98 for Magnitude, 9.348 for SparseGPT, and 9.451 for Wanda. This confirms the effectiveness of our product-based pruning metric across various methods.

TABLE VI: Ablation of pruning metric in LLaMA3-8B.

Method	Pruning Metric		
w/ LLM-Barber	$\lvert\mathbf{W}\rvert$	$\lvert\partial\mathbf{E}/\partial\mathbf{W}\rvert$	$\lvert\mathbf{W}\rvert\lvert\partial\mathbf{E}/\partial\mathbf{W}\rvert$
Magnitude	186.5	14.43	**10.98**
SparseGPT	9.544	9.457	**9.348**
Wanda	9.880	9.699	**9.451**

Bold results support the pruning metric $|\mathbf{W}||\partial\mathbf{E}/\partial\mathbf{W}|$.

Calibration Data Size. We explore how varying the size of calibration data influences the performance of LLM-Barber. Figure 4 demonstrates that as the calibration sample size increases, LLM-Barber maintains robust performance. Notably, with more than four samples, our method achieves better perplexity than SparseGPT and Wanda. LLM-Barber outperforms SparseGPT even with just a single sample, underscoring its robustness across different sample sizes.

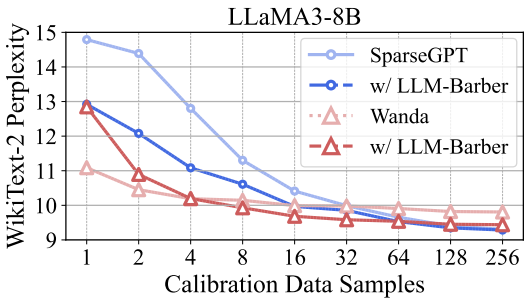

Fig. 4: Ablation of calibration size in LLaMA3-8B. LLM-Barber is robust across varying calibration size.

V. Conclusion

We propose LLM-Barber (A One-Shot Block-Aware Rebuilder for Sparsity Mask), a novel framework that rebuilds sparsity mask to optimize LLM post-training pruning. By integrating a block-aware approach across Self-Attention and MLP blocks, LLM-Barber effectively reallocates sparsity to improve accuracy without the need for extensive fine-tuning. Specifically, LLM-Barber identifies novel importance score after mask initialization and rebuilds the sparsity mask with mask rebuilding pairs, simultaneously applying new sparsity masks to weights that have become less critical, thereby optimizing overall model performance. By being the first to utlize the product of weights and gradients as a pruning metric within LLM post-training pruning, our approach enables precise and efficient reallocation of sparsity mask. Extensive experiments on pruning LLaMA series models demonstrate that LLM-Barber achieves state-of-the-art results in both perplexity and zero-shot performance in the domain of post-training pruning.

REFERENCES

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[2] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," in *5th International Conference on Learning Representations, ICLR 2017-Conference Track Proceedings*, 2019.

[3] N. Lee, T. Ajanthan, and P. Torr, "Snip: single-shot network pruning based on connection sensitivity," in *International Conference on Learning Representations*. Open Review, 2019.

[4] E. Frantar and D. Alistarh, "SparseGPT: Massive language models can be accurately pruned in one-shot," in *International Conference on Machine Learning*. PMLR, 2023, pp. 10 323–10 337.

[5] Z. Liu, H. Mu, X. Zhang, Z. Guo, X. Yang, K.-T. Cheng, and J. Sun, "Metapruning: Meta learning for automatic neural network channel pruning," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3296–3305.

[6] E. J. Hu, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, "Lora: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2021.

[7] Y. Zhang, L. Zhao, M. Lin, S. Yunyun, Y. Yao, X. Han, J. Tanner, S. Liu, and R. Ji, "Dynamic sparse no training: Training-free fine-tuning for sparse llms," in *The Twelfth International Conference on Learning Representations*, 2024.

[8] M. Sun, Z. Liu, A. Bair, and J. Z. Kolter, "A simple and effective pruning approach for large language models," in *The Twelfth International Conference on Learning Representations*, 2023.

[9] Y. LeCun, J. Denker, and S. Solla, "Optimal brain damage," *Advances in neural information processing systems*, vol. 2, 1989.

[10] B. Hassibi, D. Stork, and G. Wolff, "Optimal brain surgeon: Extensions and performance comparisons," *Advances in neural information processing systems*, vol. 6, 1993.

[11] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "GPTQ: Accurate post-training compression for generative pretrained transformers," *ICLR*, 2023.

[12] Z. Guan, H. Huang, Y. Su, H. Huang, N. Wong, and H. Yu, "APTQ: Attention-aware post-training mixed-precision quantization for large language models," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*, 2024, pp. 1–6.

[13] P. Xu, W. Shao, M. Chen, S. Tang, K. Zhang, P. Gao, F. An, Y. Qiao, and P. Luo, "Besa: Pruning large language models with blockwise parameter-efficient sparsity allocation," in *The Twelfth International Conference on Learning Representations*, 2024.

[14] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," *Advances in neural information processing systems*, vol. 28, 2015.

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

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

[17] AI@Meta, "Llama 3 model card," 2024. [Online]. Available: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md

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

[19] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of machine learning research*, vol. 21, no. 140, pp. 1–67, 2020.

[20] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," *International Conference on Learning Representations*, Sep 2017.

[21] M. Marcus, G. Kim, M. A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger, "The penn treebank," in *Proceedings of the workshop on Human Language Technology - HLT '94*, Jan 1994. [Online]. Available: http://dx.doi.org/10.3115/1075812.1075835

[22] L. Gao, J. Tow, B. Abbasi, S. Biderman, S. Black, A. DiPofi, C. Foster, L. Golding, J. Hsu, A. Le Noac'h, H. Li, K. McDonell, N. Muennighoff, C. Ociepa, J. Phang, L. Reynolds, H. Schoelkopf, A. Skowron, L. Sutawika, E. Tang, A. Thite, B. Wang, K. Wang, and A. Zou, "A framework for few-shot language model evaluation," 12 2023. [Online]. Available: https://zenodo.org/records/10256836

[23] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova, "Boolq: Exploring the surprising difficulty of natural yes/no questions," in *Proceedings of the 2019 Conference of the North*, Jan 2019. [Online]. Available: http://dx.doi.org/10.18653/v1/n19-1300

[24] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Jan 2018. [Online]. Available: http://dx.doi.org/10.18653/v1/w18-5446

[25] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, "Hellaswag: Can a machine really finish your sentence?" in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Jan 2019. [Online]. Available: http://dx.doi.org/10.18653/v1/p19-1472

[26] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, "Think you have solved question answering? try arc, the ai2 reasoning challenge," *arXiv: Artificial Intelligence,arXiv: Artificial Intelligence*, Mar 2018.

[27] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, "Can a suit of armor conduct electricity? a new dataset for open book question answering," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Jan 2018. [Online]. Available: http://dx.doi.org/10.18653/v1/d18-1260

[28] J. Choquette, W. Gandhi, O. Giroux, N. Stam, and R. Krashinsky, "Nvidia a100 tensor core gpu: Performance and innovation," *IEEE Micro*, vol. 41, no. 2, pp. 29–35, 2021.