RECAP: Training-Free Compensation for Coarse Activation Channel Pruning in Compressed LLMs

Mingyu Lee*, Akshat Ramachandran*, Tushar Krishna
Georgia Institute of Technology, Atlanta, GA
mlee864@gatech.edu, akshat.r@gatech.edu, tushar@ece.gatech.edu

***Abstract*—Sparsity is a key enabler for efficient inference in large language models (LLMs). While a wide spectrum of sparsification techniques—from unstructured to highly structured—have been explored to reduce computational overhead, they often involve trade-offs between hardware efficiency and model accuracy. Channel sparsity, in particular, is appealing due to its hardware-friendly structure compared to alternatives like structured N:M sparsity, but suffers from notable accuracy degradation, especially when applied to activations. To bridge this gap, we propose RECAP, a lightweight, training-free compensation method that mitigates the impact of channel pruning induced errors. RECAP exploits the statistics of the pruned channel as a representation of the sparsity-induced error and transfers it to the corresponding weights to compensate for the removal of the channel. Extensive experiments across diverse LLM families and benchmarks demonstrate that RECAP outperforms existing alternatives at all sparsity levels. On LLaMA3-8B, RECAP achieves approximately a 34% improvement in 0-shot BoolQ benchmark accuracy at a target sparsity ratio of 70%.**

I. Introduction

Recently, large language models (LLMs) [3], [23] have demonstrated exceptional performance across a wide range of applications. However, their practical deployment remains challenging due to the considerable model sizes and high inference costs. To enable efficient deployment on resource-constrained devices, post-training compression techniques—including pruning and quantization—have been actively explored to reduce the computational and memory demands of serving LLMs [2], [8], [12], [19].

Model pruning [2], [24] reduces memory footprint by removing ineffectual model parameters, such as individual weights/activations (unstructured) or blocks of weights/activations (structured), and storing sparse tensors in a compressed format [7]. Most existing LLM pruning techniques [2], [8], [20], [24] primarily focus on static weight pruning. Activation sparsity, which enforces input-dependent structure on the weight matrices by leveraging (or inducing) sparsity in the input activations is relatively unexplored [12]. This can be attributed to the dynamic, input-dependent and error-prone nature of activation sparsity. Furthermore, LLMs lack non-linear functions that naturally induce sparsity unlike traditional DNNs [4].

Among existing approaches exploring activation sparsity in LLMs [11], [12], TEAL [12] proposes a simple training-free method that applies magnitude-based unstructured pruning across activations to achieve high model-wide sparsity, while CATS [11] targets sparsification primarily within MLP blocks by leveraging activation sparsity through gating mechanisms, but achieves lower overall sparsity due to partial layer coverage.

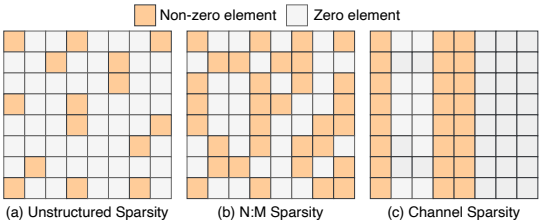

Fig. 1. Difference between (a) unstructured sparsity, (b) N:M sparsity with N=2, M=4 and (c) channel sparsity for an 8×8 representative tensor.

As demonstrated in prior work [19], [20], unstructured sparsity offers limited hardware acceleration benefits compared to more structured approaches such as N:M sparsity and channel pruning. Among these structured alternatives, channel pruning enables efficient hardware acceleration with minimal implementation complexity [9]. However, it often leads to higher compression-induced errors, posing challenges for maintaining model accuracy.

Contributions. Motivated by this challenge, we propose RECAP, a lightweight, training-free compensation method designed to mitigate the accuracy degradation caused by coarse-grained activation channel pruning. RECAP introduces an effective compensation strategy that leverages the average magnitude of pruned activation channels to scale the corresponding weight channels, thereby offsetting the error introduced by channel removal. To further enhance error correction in the presence of activation outliers—which can skew the average magnitude—we incorporate a fine-grained grouped channel strategy that captures localized activation distributions, leading to improved recovery fidelity. Extensive experiments across multiple LLM model families demonstrate that RECAP achieves up to 34% improvement over existing alternatives across a variety of benchmarks.

II. Background

A. Sparsity in LLMs

In large language models (LLMs), sparsity primarily arises from two sources: weights and activations. Weight sparsity

*Equal contribution

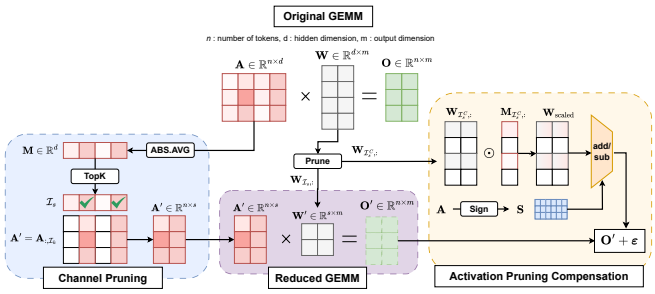

Fig. 2. Overview of RECAP. (Left) Conventional channel pruning selects channels based on magnitude, removing those with lower values. (Center) Linear Layer's GEMM operation is then reduced for efficiency. (Right) RECAP compensates the pruning error by scaling the pruned weights using the average magnitude of each channel and accumulating them based on the sign of the corresponding activation.

is typically introduced by statically eliminating redundant or ineffectual parameters, enabling model compression with minimal impact on accuracy [7], [19], [20]. In contrast, activation sparsity emerges dynamically at runtime, either as a result of non-linear operations such as ReLU [9] or by selectively pruning the outputs of inactive or low-importance neurons. The activation sparsity of a tensor A is characterized by the fraction of its entries that are zero. This sparsity can influence computation through two mechanisms [12]:

- **Input sparsity:** When evaluating $O = AW$ with $A \in \mathbb{R}^{n\times d}$ and $W \in \mathbb{R}^{d\times m}$, the rows $W[i,:]$ associated with zero-valued columns in $A[:,i]$ are effectively unused.
- **Output sparsity:** Alternatively, when computing $y = s \odot (xW)$, where $s \in \mathbb{R}^{n\times m}$ is a binary mask, the rows $W[i,:]$ corresponding to entries $s[i,:] = 0$ are disregarded [10].

Following prior work [12], [20], we exploit input sparsity.

B. Sparsification Patterns

Unstructured Sparsity. As illustrated in Figure 1(a), unstructured sparsity [4] eliminates individual weights or activations without adhering to any specific pattern. While it offers the highest compression ratio and minimal accuracy degradation, its irregular structure limits compatibility with hardware accelerators, thereby offering limited practical speedups [20].

Structured N:M Sparsity. Fine-grained N:M sparsity imposes a structured constraint by dividing weights into groups of size M and retaining at most N non-zero elements within each group (Figure 1(b)). While pruning remains unstructured within each group, the global enforcement of this pattern significantly improves hardware compatibility and enables efficient acceleration. This structured regularity strikes a balance between compression benefits and hardware-friendliness, making N:M sparsity a widely adopted strategy in modern sparse deep learning [7].

Coarse-grained Channel Sparsity. Channel sparsity treats an entire channel as an atomic unit during pruning, removing them entirely when deemed unimportant. This approach produces highly regular sparse structures, often leading to structured matrices that are well-suited for hardware acceleration [22]. However, due to its coarse-grained nature, it leads to higher accuracy degradation despite potential for superior hardware acceleration. *In this work, we aim to mitigate the accuracy degradation typically associated with coarse-grained channel sparsity, thereby achieving the best of both worlds: high hardware efficiency and strong model performance.*

III. RECAP Framework

In this section, we introduce RECAP, *a training-free activation compensation method for coarse channel pruning in compressed large language models (LLMs)*. Building upon conventional channel pruning, RECAP exploits the statistics of the pruned channel as a representation of the sparsity-induced error and transfers it to the corresponding weights to compensate for the removal of the channel. Figure 2 provides an overview of the RECAP pipeline.

Linear Layer GEMM Operation. For a Linear layer in LLMs, the General Matrix Multiplication (GEMM) operation can be expressed as:

$$\mathbf{O} = \mathbf{AW} \tag{1}$$

with activation $\mathbf{A} \in \mathbb{R}^{n\times d}$, weight $\mathbf{W} \in \mathbb{R}^{d\times m}$ and output $\mathbf{O} \in \mathbb{R}^{n\times m}$. Here, n denotes the number of tokens, d is the hidden dimension and m is the output dimension. For simplicity, we assume a batch size of 1.

Channel Pruning. To lower the computation cost of GEMM, conventional channel pruning methods [5], [17] operates on the activation matrix $\mathbf{A}$ by first computing the average of the absolute values for each channel. A top-k selection is then applied to retain only the most significant channels. This process can be formally expressed as:

$$\mathbf{M} = \frac{1}{n}\sum_{i=0}^{n-1} |\mathbf{A}_{i,:}| \tag{2}$$

$$\mathcal{I}_s = \text{TopK}(\mathbf{M}, s) \tag{3}$$

where vector $\mathbf{M} \in \mathbb{R}^d$ is the average of each channel's magnitude and $\mathcal{I}_s$ is the indices of selected top s channels. Once $\mathcal{I}_s$ is obtained, its complement list $\mathcal{I}_s^C$ is used to prune the corresponding activation and weight channels as follows,

$$\mathbf{A}' = \mathbf{A}_{:,\mathcal{I}_s}, \quad \mathbf{W}' = \mathbf{W}_{\mathcal{I}_s,:} \tag{4}$$

where $\mathbf{A}' \in \mathbb{R}^{n\times s}$ and $\mathbf{W}' \in \mathbb{R}^{s\times m}$ represent the pruned activation and weight matrices, respectively, with $s << d$.

Reduced Linear GEMM Operation. After pruning, the GEMM computation is approximated by using only the selected top s channels. The resulting operation becomes:

$$\mathbf{O} \approx \mathbf{O}' = \mathbf{A}'\mathbf{W}' = \mathbf{A}_{:,\mathcal{I}s}\mathbf{W}_{\mathcal{I}_s,:} \tag{5}$$

This reduces the computational complexity from $O(n\cdot d\cdot m)$ to $O(n\cdot s\cdot m)$, since $s << d$, this significantly improves GEMM efficiency.

Induced Error. Although activations pruning is performed based on selecting the channels with highest average magnitude, it inevitably introduces an error due to the removal of channels. The reduced GEMM operation can be expressed in relation to the original GEMM as follows:

$$\mathbf{AW} = \mathbf{A}'\mathbf{W}' + \varepsilon + \delta \tag{6}$$

where $\varepsilon + \delta$ denotes the total error resulting from channel pruning. In this formulation, δ refers to the irrecoverable error, while ε denotes the recoverable portion. The objective of RECAP is to maximize compensation for the recoverable error ε, thereby mitigating the impact of the irrecoverable error δ.

RECAP's Compensation. RECAP compensates ε by leveraging the precomputed statistics of the pruned channel which is the average of each channel's magnitude, $\mathbf{M}_{\mathcal{I}_s^C}$. With vector $\mathbf{M}_{\mathcal{I}_s^C}$, we scale the corresponding pruned rows of the weight matrix through element-wise multiplication as:

$$\mathbf{W}_{\text{scaled}} = \mathbf{M}_{\mathcal{I}_s^C} \odot \mathbf{W}_{\mathcal{I}_s^C,:} \tag{7}$$

Here, $\odot$ represents element-wise multiplication, with broadcasting of the vector $\mathbf{M}_{\mathcal{I}_k^C}$ over to the rows of $\mathbf{W}_{\mathcal{I}_k^C,:}$. Simultaneously, sign bits of the activation matrix $\mathbf{A}$ are extracted to guide the compensation process. These sign values determine whether each scaled weight is added or subtracted. It primarily functions to prevent nullification of the compensation. The compensated error ε is then computed by accumulating the scaled weights accordingly:

$$\mathbf{S} = \text{sign}(\mathbf{A}) \tag{8}$$

$$\varepsilon = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} \sum_{k \in \mathcal{I}_s^C} \mathbf{S}_{i,k} \cdot \mathbf{W}_{\text{scaled}}^{(k,j)} \tag{9}$$

where $\varepsilon \in \mathbb{R}^{n \times m}$ is the compensation tensor and $\mathbf{W}_{\text{scaled}}^{(k,j)}$ denotes the (k, j) element of the scaled weight matrix.

Fine-grained Compensation. To better approximate ε, we propose a fine-grained grouped channel compensation strategy. This approach improves the accuracy of restoration by computing the average magnitude $\mathbf{M}$ at a finer granularity, thereby enabling better approximation even under skewed channel value distributions. This skewed channel distribution typically arises from the varied distribution of inliers and outliers in channels.

In this strategy, each channel to be compensated is divided into groups of size $\mathbf{N}_g$ along the token dimension, and per-group statistics are computed within the pruned channel subset. The formulation is as follows:

$$\mathbf{n}_g = n / \mathbf{N}_g \tag{10}$$

$$\mathbf{M}_g = \frac{1}{\mathbf{n}_g} \sum_{i=\mathbf{n}_g \cdot g}^{\mathbf{n}_g \cdot g + \mathbf{n}_g - 1} |\mathbf{A}_{i,:}| \tag{11}$$

where $\mathbf{M}_g$ denotes the average magnitude for group g within the channels.

$$\mathbf{W}_{\text{scaled}}^{(g)} = \mathbf{M}_{\mathcal{I}_s^C}^{(g)} \odot \mathbf{W}_{\mathcal{I}_s^C,:} \tag{12}$$

$$\varepsilon_g = \sum_{i=\mathbf{n}_g \cdot g}^{\mathbf{n}_g \cdot g + \mathbf{n}_g - 1} \sum_{j=0}^{m-1} \sum_{k \in \mathcal{I}_s^C} \mathbf{S}_{i,k} \cdot \mathbf{W}_{\text{scaled}}^{(g,k,j)} \tag{13}$$

$$\varepsilon = \text{concat}(\varepsilon_0, \varepsilon_1, ..., \varepsilon_{\mathbf{N}_g - 1}) \tag{14}$$

TABLE I
COMPARISON OF UNIFORM PRUNING ON PERPLEXITY, ZERO SHOT BOOLQ, 5 SHOT MMLU FOR LLAMA2-7B [23]/LLAMA3-8B [3]/QWEN2.5-3B [18]. THE RECAP GROUP SIZE IS SET TO $\mathbf{N}_g = 8$.

Model	Sparsity	Method	PPL(↓)	BoolQ(↑)	MMLU(↑)
LLaMA2-7B [23]	Baseline	-	5.47	77.74	45.81
	30%	Channel Pruning	57.95	62.29	25.42
		RECAP (Ours)	9.80	72.72	32.07
	50%	Channel Pruning	$1.65 \cdot 10^4$	48.10	23.87
		RECAP (Ours)	16.69	69.94	26.82
	70%	Channel Pruning	$9.25 \cdot 10^3$	43.70	23.72
		RECAP (Ours)	72.30	67.09	25.17
LLaMA3-8B [3]	Baseline	-	6.24	82.08	65.41
	30%	Channel Pruning	192.51	62.20	26.19
		RECAP (Ours)	29.32	78.44	40.50
	50%	Channel Pruning	$8.50 \cdot 10^3$	42.32	26.04
		RECAP (Ours)	33.46	78.35	39.25
	70%	Channel Pruning	$3.85 \cdot 10^4$	38.75	24.72
		RECAP (Ours)	94.29	72.48	27.91
Qwen2.5-3B [18]	Baseline	-	8.03	77.10	65.61
	30%	Channel Pruning	50.36	61.23	26.24
		RECAP (Ours)	6.43	74.36	49.51
	50%	Channel Pruning	$1.26 \cdot 10^3$	53.37	25.17
		RECAP (Ours)	10.36	70.26	47.85
	70%	Channel Pruning	$3.19 \cdot 10^4$	47.31	23.58
		RECAP (Ours)	37.35	66.16	33.65

(50% Pruning Level, LLaMA2-7B)
Input Text : The largest animal
Baseline : is the blue whale. It can grow to 30m in length and weigh over ..
Channel : is the elephant, which weighs up to 6 tons and can reach 25 feet..
RECAP : is the blue whale, which can reach 30 meters (98 ft) in length and..

Input Text : A Large Language Model is
Baseline : a type of deep learning model that has been trained on a ..
Channel : a language that is used by a large number of people..
RECAP : a type of artificial intelligence (AI) model that is capable of..

Fig. 3. Text generation examples from LLaMA2-7B under 50% sparsity ratio. Incorrect outputs and correct outputs are highlighted.

Here, $\mathbf{W}_{\text{scaled}}^{(g)}$ represents the scaled weights for group g. The term $\varepsilon_g \in \mathbb{R}^{\mathbf{n}_g \times m}$ corresponds to the compensation tensor for group g, aggregated across spatial and channel dimensions. Finally, the complete compensation tensor ε is formed by concatenating the group-wise compensation tensors across all $\mathbf{N}_g$ groups.

RECAP transfers the statistical information of pruned channels to their corresponding weights, transforming full matrix multiplications into lightweight operations comprising element-wise scaling and sign-guided accumulation. This design enables effective ε compensation with minimal computational overhead.

IV. EXPERIMENTAL EVALUATIONS

A. Experimental Setup

Models and Datasets. We evaluate RECAP on LLM families LLaMA2-7B [23], LLaMA3-8B [3] and Qwen2.5-3B [18]. Each model's perplexity is measured with WikiText2 [16] dataset and benchmark accuracy on BoolQ [1], MMLU [6] and HellaSwag [25] (ablations), under both conventional channel sparsity without compensation and the proposed RECAP compensation.

Implementation Details. RECAP is implemented in PyTorch and all experiments are conducted on a single NVIDIA GH200 GPU. We set the group size as $\mathbf{N_g} = 8$, which is empirically determined. RECAP is applied to activations in linear layers,

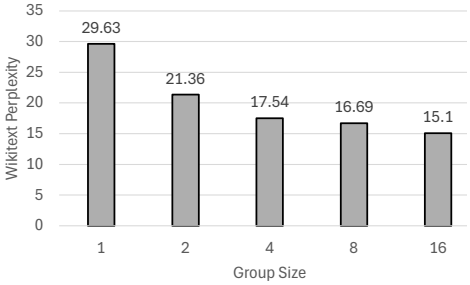

Fig. 4. Effect of group size ($\mathbf{N_g}$) on Wikitext perplexity for LLaMA2-7B [23] at uniform sparsity of 50%.

TABLE II
NON-UNIFORM LAYER-WISE PRUNING COMPARISON BETWEEN TEAL [12] AND RECAP FOR LLAMA3-8B [3] AT A TARGET SPARSITY RATIO OF 60%.

Method	BoolQ (↑)	HellaSwag (↑)	Average (↑)
Baseline	82.08	82.20	82.14
TEAL [12]	73.26	69.85	71.56
RECAP (Ours)	78.36	74.63	76.50

targeting query, key, value, attention output projections and feed-forward layers inside the transformer block.

B. Perplexity Evaluation

In Table I, we compare the perplexity results of WikiText2 on LLaMA2-7B, LLaMA3-8B and Qwen2.5-3B across 30%/50%/70% uniform pruning. RECAP consistently achieves substantially lower perplexity compared to channel pruning across all sparsity ratio and models, demonstrating it's effectiveness in recovering the error ε. Notably, under 50% sparsity ratio, RECAP reduces the perplexity by up to $3.84 \cdot 10^4$, showing its robustness in maintaining model performance even under aggressive pruning.

C. Downstream Task Evaluation

As presented in Table I, RECAP consistently recovers a substantial portion of the lost accuracy from channel pruning across BoolQ and MMLU benchmarks, highlighting its robustness across a variety of tasks and architectures. Notably, RECAP restores up to 34% accuracy on LLaMA3-8B for BoolQ and 23% accuracy on Qwen2.5-3B for MMLU compared to conventional channel pruning.

D. Text Generation

In Figure 3, we compare the generated sentences with naive channel pruning and RECAP with Llama2-7B model under 50% activation pruning. Compared to channel pruning without compensation, RECAP significantly improves output quality, effectively mitigating semantic distortions caused by pruning. For instance, when asked about the largest animal, channel pruning produces an incorrect output referring to elephants, whereas RECAP correctly identifies the blue whale, demonstrating its ability to accurately recover lost information.

E. Ablations and Discussions

Effect of group size $\mathbf{N}_g$. We illustrate the impact of varying $\mathbf{N}_g$ values on model perplexity in Figure 4. As observed, increasing $\mathbf{N}_g$ consistently leads to lower perplexity, attributed to improved error approximation at finer granularity, which better captures skewed channel distributions caused by outliers. However, setting $\mathbf{N}_g$ too large degrades inference performance due to increased computational overhead. To balance accuracy and efficiency tradeoff, we select $\mathbf{N}_g = 8$.

Non-uniform Layer-wise Pruning. Since TEAL [12] applies non-uniform layer-wise pruning using a greedy optimization, we employ FLOW [20] to determine optimal layerwise sparsity ratios for RECAP, ensuring a fair comparison at a target sparsity of 60%. Both methods use a calibration set of 256 samples drawn from WikiText [16]. As shown in Table II, RECAP achieves approximately 5% higher benchmark accuracy than TEAL, highlighting the effectiveness of RECAP's compensation for pruning-induced errors.

Future Directions. Our technique is orthogonal to existing weight compression methods and can be integrated alongside them to enable further model compression, which we plan to investigate this integration in future work. Additionally, we plan to theoretically validate our method and also develop a formulation for the unrecoverable error component δ.

V. RELATED WORK

LLM Pruning. Recent advances in unstructured pruning, such as SparseGPT [2], leveraged the inverse Hessian for importance estimation, while Wanda [21] proposed a simple criterion based on the product of weight magnitudes and activations. Extensions of these techniques to structured N:M sparsity have largely been restricted to applying a fixed N:M pattern uniformly across all layers. However, works such as [20], [24] emphasized the need for heterogeneous sparsity budgets across different layers, advocating for a non-uniform, layer-wise N:M sparsity assignment based on the presence and distribution of outliers. In parallel, recent research [12] explored magnitude pruning of LLM activations, motivated by the observation that activation distributions are typically zero-mean and unimodal.

Error Compensation. To compensate for the sparsity induced errors, early techniques as [15] are not practical for large LLMs due to the demand for significant compute resources required for model finetuning. To overcome this challenge, techniques such as [2] leverage the inverse Hessian to compensate for sparsification errors, [14] explored parameter-efficient finetuning for sparsified LLMs, [13] projects compression errors into a low-rank space, and minimizes compression-induced errors without requiring gradient-based training.

VI. CONCLUSION

We propose RECAP, a lightweight, training-free compensation method to mitigate the impact of channel pruning errors. Utilizing the statistics of pruned activation channels, RECAP transfers the approximation to corresponding weights and effectively compensates pruning induced errors. Extensive experiments across various LLM families demonstrate that RECAP achieve substantial error recovery, consistently improving benchmark accuracy and perplexity. Our approach

highlights the potential of RECAP to bridge the gap between hardware efficiency and model performance in the context of pruning, offering a promising future direction of exploration.

ACKNOWLEDGMENTS

This work was supported in part by CoCoSys, one of the seven centers in JUMP 2.0, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.

REFERENCES

[1] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova, "Boolq: Exploring the surprising difficulty of natural yes/no questions," 2019. [Online]. Available: https://arxiv.org/abs/1905.10044

[2] E. Frantar and D. Alistarh, "SparseGPT: Massive language models can be accurately pruned in one-shot," 2023.

[3] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, A. Yang, A. Fan, A. Goyal, A. Hartshorn, A. Yang, A. Mitra, A. Sravankumar, A. Korenev, A. Hinsvark, A. Rao, A. Zhang, A. Rodriguez, A. Gregerson, A. Spataru, B. Roziere, B. Biron, B. Tang, B. Chern, C. Caucheteux, C. Nayak, C. Bi, C. Marra, C. McConnell, C. Keller, C. Touret, C. Wu, C. Wong, C. C. Ferrer, C. Nikolaidis, D. Allonsius, D. Song, D. Pintz, D. Livshits, D. Wyatt, D. Esiobu, D. Choudhary, D. Mahajan, D. Garcia-Olano, D. Perino, D. Hupkes, E. Lakomkin, E. AlBadawy, E. Lobanova, E. Dinan, E. M. Smith, F. Radenovic, F. Guzmán, F. Zhang, G. Synnaeve, G. Lee, G. L. Anderson, G. Thattai, G. Nail, G. Mialon, G. Pang, G. Cucurell, H. Nguyen, H. Korevaar, H. Xu, H. Touvron, I. Zarov, I. A. Ibarra, I. Kloumann, I. Misra, I. Evtimov, J. Zhang, J. Copet, J. Lee, J. Geffert, J. Vranes, J. Park, J. Mahadeokar, J. Shah, J. van der Linde, J. Billock, J. Hong, J. Lee, J. Fu, J. Chi, J. Huang, J. Liu, J. Wang, J. Yu, J. Bitton, J. Spisak, J. Park, J. Rocca, J. Johnstun, J. Saxe, J. Jia, K. V. Alwala, K. Prasad, K. Upasani, K. Plawiak, K. Li, K. Heafield, K. Stone, K. El-Arini, K. Iyer, K. Malik, K. Chiu, K. Bhalla, K. Lakhotia, L. Rantala-Yeary, L. van der Maaten, L. Chen, L. Tan, L. Jenkins, L. Martin, L. Madaan, L. Malo, L. Blecher, L. Landzaat, L. de Oliveira, M. Muzzi, M. Pasupuleti, M. Singh, M. Paluri, M. Kardas, M. Tsimpoukelli, M. Oldham, M. Rita, M. Pavlova, M. Kambadur, M. Lewis, M. Si, M. K. Singh, M. Hassan, N. Goyal, N. Torabi, N. Bashlykov, N. Bogoychev, N. Chatterji, N. Zhang, O. Duchenne, O. Çelebi, P. Alrassy, P. Zhang, P. Li, P. Vasic, P. Weng, P. Bhargava, P. Dubal, P. Krishnan, P. S. Koura, P. Xu, Q. He, Q. Dong, R. Srinivasan, R. Ganapathy, R. Calderer, R. S. Cabral, R. Stojnic, R. Raileanu, R. Maheswari, R. Girdhar, R. Patel, R. Sauvestre, R. Polidoro, R. Sumbaly, R. Taylor, R. Silva, R. Hou, R. Wang, S. Hosseini, S. Chennabasappa, S. Singh, S. Bell, S. S. Kim, S. Edunov, S. Nie, S. Narang, S. Raparthy, S. Shen, S. Wan, S. Bhosale, S. Zhang, S. Vandenhende, S. Batra, S. Whitman, S. Sootla, S. Collot, S. Gururangan, S. Borodinsky, T. Herman, T. Fowler, T. Sheasha, T. Georgiou, T. Scialom, T. Speckbacher, T. Mihaylov, T. Xiao, U. Karn, V. Goswami, V. Gupta, V. Ramanathan, V. Kerkez, V. Gonguet, V. Do, V. Vogeti, V. Albiero, V. Petrovic, W. Chu, W. Xiong, W. Fu, W. Meers, X. Martinet, X. Wang, X. Wang, X. E. Tan, X. Xia, X. Xie, X. Jia, X. Wang, Y. Goldschlag, Y. Gaur, Y. Babaei, Y. Wen, Y. Song, Y. Zhang, Y. Li, Y. Mao, Z. D. Coudert, Z. Yan, Z. Chen, Z. Papakipos, A. Singh, A. Srivastava, A. Jain, A. Kelsey, A. Shajnfeld, A. Gangidi, A. Victoria, A. Goldstand, A. Menon, A. Sharma, A. Boesenberg, A. Baevski, A. Feinstein, A. Kallet, A. Sangani, A. Teo, A. Yunus, A. Lupu, A. Alvarado, A. Caples, A. Gu, A. Ho, A. Poulton, A. Ryan, A. Ramchandani, A. Dong, A. Franco, A. Goyal, A. Saraf, A. Chowdhury, A. Gabriel, A. Bharambe, A. Eisenman, A. Yazdan, B. James, B. Maurer, B. Leonhardi, B. Huang, B. Loyd, B. D. Paola, B. Paranjape, B. Liu, B. Wu, B. Ni, B. Hancock, B. Wasti, B. Spence, B. Stojkovic, B. Gamido, B. Montalvo, C. Parker, C. Burton, C. Mejia, C. Liu, C. Wang, C. Kim, C. Zhou, C. Hu, C.-H. Chu, C. Cai, C. Tindal, C. Feichtenhofer, C. Gao, D. Civin, D. Beaty, D. Kreymer, D. Li, D. Adkins, D. Xu, D. Testuggine, D. David, D. Parikh, D. Liskovich, D. Foss, D. Wang, D. Le, D. Holland, E. Dowling, E. Jamil, E. Montgomery, E. Presani, E. Hahn, E. Wood, E.-T. Le, E. Brinkman, E. Arcaute, E. Dunbar, E. Smothers, F. Sun, F. Kreuk, F. Tian, F. Kokkinos, F. Ozgenel, F. Caggioni, F. Kanayet, F. Seide, G. M. Florez, G. Schwarz, G. Badeer, G. Swee, G. Halpern, G. Herman, G. Sizov, Guangyi, Zhang, G. Lakshminarayanan, H. Inan, H. Shojanazeri, H. Zou, H. Wang, H. Zha, H. Habeeb, H. Rudolph, H. Suk, H. Aspegren, H. Goldman, H. Zhan, I. Damlaj, I. Molybog, I. Tufanov, I. Leontiadis, I.-E. Veliche, I. Gat, J. Weissman, J. Geboski, J. Kohli, J. Lam, J. Asher, J.-B. Gaya, J. Marcus, J. Tang, J. Chan, J. Zhen, J. Reizenstein, J. Teboul, J. Zhong, J. Jin, J. Yang, J. Cummings, J. Carvill, J. Shepard, J. McPhie, J. Torres, J. Ginsburg, J. Wang, K. Wu, K. H. U, K. Saxena, K. Khandelwal, K. Zand, K. Matosich, K. Veeraraghavan, K. Michelena, K. Li, K. Jagadeesh, K. Huang, K. Chawla, K. Huang, L. Chen, L. Garg, L. A, L. Silva, L. Bell, L. Zhang, L. Guo, L. Yu, L. Moshkovich, L. Wehrstedt, M. Khabsa, M. Avalani, M. Bhatt, M. Mankus, M. Hasson, M. Lennie, M. Reso, M. Groshev, M. Naumov, M. Lathi, M. Keneally, M. Liu, M. L. Seltzer, M. Valko, M. Restrepo, M. Patel, M. Vyatskov, M. Samvelyan, M. Clark, M. Macey, M. Wang, M. J. Hermoso, M. Metanat, M. Rastegari, M. Bansal, N. Santhanam, N. Parks, N. White, N. Bawa, N. Singhal, N. Egebo, N. Usunier, N. Mehta, N. P. Laptev, N. Dong, N. Cheng, O. Chernoguz, O. Hart, O. Salpekar, O. Kalinli, P. Kent, P. Parekh, P. Saab, P. Balaji, P. Rittner, P. Bontrager, P. Roux, P. Dollar, P. Zvyagina, P. Ratanchandani, P. Yuvraj, Q. Liang, R. Alao, R. Rodriguez, R. Ayub, R. Murthy, R. Nayani, R. Mitra, R. Parthasarathy, R. Li, R. Hogan, R. Battey, R. Wang, R. Howes, R. Rinott, S. Mehta, S. Siby, S. J. Bondu, S. Datta, S. Chugh, S. Hunt, S. Dhillon, S. Sidorov, S. Pan, S. Mahajan, S. Verma, S. Yamamoto, S. Ramaswamy, S. Lindsay, S. Lindsay, S. Feng, S. Lin, S. C. Zha, S. Patil, S. Shankar, S. Zhang, S. Zhang, S. Wang, S. Agarwal, S. Sajuyigbe, S. Chintala, S. Max, S. Chen, S. Kehoe, S. Satterfield, S. Govindaprasad, S. Gupta, S. Deng, S. Cho, S. Virk, S. Subramanian, S. Choudhury, S. Goldman, T. Remez, T. Glaser, T. Best, T. Koehler, T. Robinson, T. Li, T. Zhang, T. Matthews, T. Chou, T. Shaked, V. Vontimitta, V. Ajayi, V. Montanez, V. Mohan, V. S. Kumar, V. Mangla, V. Ionescu, V. Poenaru, V. T. Mihailescu, V. Ivanov, W. Li, W. Wang, W. Jiang, W. Bouaziz, W. Constable, X. Tang, X. Wu, X. Wang, X. Wu, X. Gao, Y. Kleinman, Y. Chen, Y. Hu, Y. Jia, Y. Qi, Y. Li, Y. Zhang, Y. Zhang, Y. Adi, Y. Nam, Yu, Wang, Y. Zhao, Y. Hao, Y. Qian, Y. Li, Y. He, Z. Rait, Z. DeVito, Z. Rosnbrick, Z. Wen, Z. Yang, Z. Zhao, and Z. Ma, "The llama 3 herd of models," 2024. [Online]. Available: https://arxiv.org/abs/2407.21783

[4] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[5] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," 2017. [Online]. Available: https://arxiv.org/abs/1707.06168

[6] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring massive multitask language understanding," 2021. [Online]. Available: https://arxiv.org/abs/2009.03300

[7] G. Jeong, S. Damani, A. R. Bambhaniya, E. Qin, C. J. Hughes, S. Subramoney, H. Kim, and T. Krishna, "Vegeta: Vertically-integrated extensions for sparse/dense gemm tile acceleration on cpus," in *2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 2023, pp. 259–272.

[8] G. Jeong, P.-A. Tsai, S. W. Keckler, and T. Krishna, "Sdq: Sparse decomposed quantization for llm inference," *arXiv preprint arXiv:2406.13868*, 2024.

[9] M. Kurtz, J. Kopinsky, R. Gelashvili, A. Matveev, J. Carr, M. Goin, W. Leiserson, S. Moore, N. Shavit, and D. Alistarh, "Inducing and exploiting activation sparsity for fast inference on deep neural networks," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5533–5543.

[10] D. Lee, J. Lee, G. Zhang, M. Tiwari, and A. Mirhoseini, "Cats: Context-aware thresholding for sparsity in large language models," in *First Conference on Language Modeling*.

[11] D. Lee, J.-Y. Lee, G. Zhang, M. Tiwari, and A. Mirhoseini, "Cats: Contextually-aware thresholding for sparsity in large language models," *arXiv preprint arXiv:2404.08763*, 2024.

[12] J. Liu, P. Ponnusamy, T. Cai, H. Guo, Y. Kim, and B. Athiwaratkun, "Training-free activation sparsity in large language models," *arXiv preprint arXiv:2408.14690*, 2024.

[13] S.-Y. Liu, M. Khadkevich, N. C. Fung, C. Sakr, C.-H. H. Yang, C.-Y. Wang, S. Muralidharan, H. Yin, K.-T. Cheng, J. Kautz *et al.*, "Eora:

Training-free compensation for compressed llm with eigenspace low-rank approximation," *arXiv preprint arXiv:2410.21271*, 2024.

[14] X. Lu, A. Zhou, Y. Xu, R. Zhang, P. Gao, and H. Li, "Spp: Sparsity-preserved parameter-efficient fine-tuning for large language models," *arXiv preprint arXiv:2405.16057*, 2024.

[15] X. Ma, G. Fang, and X. Wang, "Llm-pruner: On the structural pruning of large language models," *Advances in neural information processing systems*, vol. 36, pp. 21 702–21 720, 2023.

[16] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," 2016. [Online]. Available: https://arxiv.org/abs/1609.07843

[17] C. Park, M. Park, H. J. Oh, M. Kim, M. K. Yoon, S. Kim, and W. W. Ro, "Balanced column-wise block pruning for maximizing gpu parallelism," in *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, ser. AAAI'23/IAAI'23/EAAI'23. AAAI Press, 2023. [Online]. Available: https://doi.org/10.1609/aaai.v37i8.26126

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

[19] A. Ramachandran, S. Kundu, and T. Krishna, "Microscopiq: Accelerating foundational models through outlier-aware microscaling quantization," *arXiv preprint arXiv:2411.05282*, 2024.

[20] A. Ramachandran, S. Kundu, A. Raha, S. Kundu, D. K. Mathaikutty, and T. Krishna, "Accelerating llm inference with flexible n: M sparsity via a fully digital compute-in-memory accelerator," *arXiv preprint arXiv:2504.14365*, 2025.

[21] M. Sun, Z. Liu, A. Bair, and J. Z. Kolter, "A simple and effective pruning approach for large language models," *ICLR*, 2024.

[22] W. Sun, A. Zhou, S. Stuijk, R. Wijnhoven, A. O. Nelson, H. Corporaal *et al.*, "Dominosearch: Find layer-wise fine-grained n: M sparse schemes from dense neural networks," *Advances in neural information processing systems*, vol. 34, pp. 20 721–20 732, 2021.

[23] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom, "Llama 2: Open foundation and fine-tuned chat models," 2023. [Online]. Available: https://arxiv.org/abs/2307.09288

[24] L. Yin, Y. Wu, Z. Zhang, C.-Y. Hsieh, Y. Wang, Y. Jia, G. Li, A. Jaiswal, M. Pechenizkiy, Y. Liang *et al.*, "Outlier weighed layerwise sparsity: A missing secret sauce for pruning llms to high sparsity," *ICML*, 2024.

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