Lightweight Haar Wavelet Subband Pruning for LLMs

Jiang Li[1,3], **Pengfei Cao**[2], **Chenxi Zhou**[2], **Tian Lan**[1,3], **Xiangdong Su**[1,3]*, **Kang Liu**[2], **Jun Zhao**[2], **Guanglai Gao**[1,3]

[1] College of Computer Science, Inner Mongolia University, China

[2] The Key Laboratory of Cognition and Decision Intelligence for Complex Systems, Institute of Automation, Chinese Academy of Sciences

[3] National & Local Joint Engineering Research Center of Intelligent Information Processing Technology for Mongolian, China

lijiangimu@gmail.com,cssxd@imu.edu.cn

Abstract

Large language models (LLMs) reach state-of-the-art performance across many NLP tasks, but their large parameter counts introduce heavy computational and memory overhead, which complicates deployment in resource-constrained settings. Pruning is a standard compression strategy that induces sparsity to lower these costs. However, most pruning methods for LLMs depend on calibration data and expensive weight updates, which limits practical scalability. To address these limitations, we introduce **H**aar **W**avelet **S**ubband **P**runing (HWSP), a post-training framework that requires no calibration data and no weight updates. HWSP applies a two-dimensional Haar wavelet transform to each weight matrix and decomposes it into four frequency subbands. It then assigns a uniform sparsity ratio to all subbands so that both low- and high-frequency components are retained in a balanced manner. Our theoretical analysis shows that the subband design of HWSP provides a deterministic per-subband retention guarantee, which helps mitigate the potential bias of global magnitude pruning toward dominant frequency components. Experiments on the LLaMA, OPT and Qwen model families show that HWSP achieves competitive accuracy relative to strong pruning baselines while substantially reducing pruning time. Compared with magnitude pruning, which serves as a simple calibration-free baseline, HWSP generally achieves better downstream performance across a wide range of sparsity levels and model scales.[1].

1 Introduction

Large language models (LLMs) (Achiam et al., 2023; Touvron et al., 2023; Yang et al., 2024a) have achieved impressive performance across complex natural language tasks. However, their large parameter counts demand substantial computational and memory resources, limiting deployment in low-resource settings. To alleviate these costs, model pruning has emerged as an effective strategy (Frantar and Alistarh, 2023; Sun et al., 2024; Bai et al., 2024), selectively removing less important weights to reduce memory and computation while preserving model architecture.

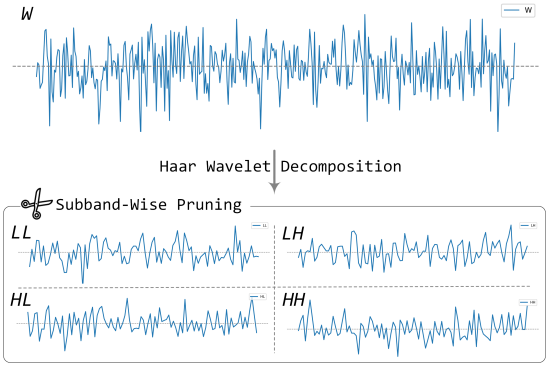


Figure 1: An illustration of HWSP.

Early pruning approaches (LeCun et al., 1989; Hassibi et al., 1993) relied on second-order information derived from the Hessian matrix to identify and remove redundant weights. Later work (Hoefler et al., 2021) demonstrated pruning's empirical success in smaller-scale models across vision and language tasks. However, applying these techniques to LLMs remains challenging due to the high cost of retraining billions of parameters after pruning (Zhao et al., 2024). To address this, recent methods (Frantar and Alistarh, 2023; Sun et al., 2024; Bai et al., 2024) have focused on post-training pruning that avoids full retraining. SparseGPT (Frantar and Alistarh, 2023) incorporated second-order information and activation statistics collected from a small calibration dataset, enabling pruning without retraining. Wanda (Sun et al., 2024) further improved efficiency by combining weight magnitude with activation values to identify pruning candidates, thus removing the

*Corresponding Author

[1]Code is available at https://github.com/dellixx/HWSP

Findings of the Association for Computational Linguistics: ACL 2026, pages 16242–16259
July 2-7, 2026

need for weight updates. Despite these advances, two critical challenges remain. **First**, most effective pruning methods relied heavily on calibration data to estimate parameter importance. However, as shown by Ji et al. (Ji et al., 2025), pruning performance is highly sensitive to the choice of calibration dataset. Moreover, increasing the quantity of calibration data often fails to compensate for poor data quality or domain mismatch, severely limiting generalization in practical deployment scenarios. **Second**, methods that do not rely on calibration data or weight updates, such as pure magnitude pruning (Han et al., 2015b), typically suffer from substantial performance degradation at moderate sparsity levels. For example, SparseGPT and Wanda both report sharp accuracy drops beyond a moderate sparsity level (around 30%) when using magnitude pruning alone, making it unsuitable for aggressive compression.

In summary, recent methods have enabled pruning of billion-scale LLMs without full retraining, yet still rely on calibration data or expensive weight updates. However, both approaches introduce considerable computational overhead during pruning. In contrast, magnitude pruning is directly based on weight magnitude rankings and avoids expensive computation. However, such methods often fail to maintain accuracy at moderate sparsity levels. Therefore, maintaining high model performance under moderate sparsity without incurring substantial computational cost remains a key challenge for practical deployment.

To overcome the above limitations, this paper proposes Haar wavelet subband pruning (HWSP), a lightweight pruning framework that requires neither weight updates nor calibration data. The core motivation for using the Haar wavelet transform in HWSP lies in its ability to expose the multi-scale structure of weight matrices. Because the Haar transform is an orthonormal and invertible linear operator, the squared ℓ_2 reconstruction error incurred by removing a coefficient equals the square of that coefficient's magnitude. Hence, error estimation depends solely on the coefficients themselves, which allows weight importance to be ranked directly in the frequency domain without relying on activation sampling or calibration data. Specifically, HWSP decomposes a weight matrix into four frequency subbands: LL, representing a downsampled low-frequency approximation, and LH, HL, and HH, which capture high-frequency variations along horizontal, vertical, and diagonal directions, respectively, as shown in Figure 1. This decomposition enables a more fine-grained analysis of the structural information encoded in the weights. HWSP first applies the Haar transform to each weight matrix, decomposing it into four frequency subbands. Then, all subbands are pruned with the same sparsity ratio. Unlike global magnitude pruning, which tends to retain more low-frequency coefficients due to their larger magnitudes while discarding high-frequency components, HWSP ensures balanced pruning across the spectrum. This strategy helps preserve high-frequency coefficients that often encode important structural details. We evaluate HWSP on the LLaMA, OPT, and Qwen model families, and demonstrate that it achieves accuracy comparable to existing methods such as SparseGPT and Wanda, while substantially improving pruning efficiency. Furthermore, compared with magnitude pruning, a widely adopted calibration-free baseline, HWSP generally achieves better performance across a range of model architectures and sparsity settings.

In summary, this paper has the following contributions:

- We propose **H**aar **W**avelet **S**ubband **P**runing (HWSP), a lightweight method for post-training sparsification of LLMs. It requires neither calibration data nor weight updates and is suitable for deployment in resource-constrained settings.

- We develop a frequency-aware strategy that applies a two-dimensional Haar transform to each weight matrix, partitions coefficients into four subbands, and enforces uniform sparsity across subbands. This allocation mitigates the low-frequency bias of global magnitude pruning and preserves localized high-frequency structure to maintain expressiveness.

- We conduct comprehensive experiments on the LLaMA, OPT and Qwen families and show that HWSP matches the accuracy of state-of-the-art pruning baselines while substantially reducing pruning time. Relative to calibration-free magnitude pruning, HWSP yields consistently stronger results in most evaluated settings.

Due to space limitations, the discussion of **Related Work** is provided in Appendix A.

2 Methodology

In this section, we introduce a Haar wavelet subband pruning framework for LLMs, which leverages the spatial and frequency-domain structure inherent in Transformer weight matrices. The approach begins by applying a two-dimensional Haar wavelet transform to each weight matrix, decomposing it into frequency subbands. A uniform sparsity ratio is then independently applied to each subband. Finally, the pruned weight matrix is reconstructed via the inverse wavelet transform. This subband-wise strategy promotes balanced sparsification across different frequency components, facilitating the retention of both global and local structural information. The proposed method, HWSP, enables efficient post-training pruning without relying on calibration data or weight updates.

2.1 HWSP Method

Haar Wavelet Decomposition. Let $\mathbf{W} \in \mathbb{R}^{m\times n}$ denote a weight matrix from a Transformer block. We treat $\mathbf{W}$ as a two-dimensional discrete signal and apply the orthonormal 2D Haar wavelet transform HWT to obtain four subbands representing different frequency components:

$$[\mathbf{LL}, (\mathbf{LH}, \mathbf{HL}, \mathbf{HH})] = \mathrm{HWT}(\mathbf{W}), \quad (1)$$

where $\mathbf{LL}$ denotes the approximation (low-low) coefficients, while $\mathbf{LH}$, $\mathbf{HL}$, and $\mathbf{HH}$ represent the horizontal, vertical, and diagonal detail coefficients, respectively. Each subband lies in $\mathbb{R}^{\frac{m}{2}\times\frac{n}{2}}$. The function $\mathrm{HWT}(\cdot)$ performs a two-dimensional Haar wavelet transform using local averaging and differencing over 2×2 patches, effectively separating low- and high-frequency components. In implementation, we first reshape $\mathbf{W}$ to a tensor of shape $(1, 1, m, n)$, treating it as a single-channel image. The transform is then applied via a stride-2 convolution, and the resulting subbands are reshaped back to their original dimensions. This operation enables frequency-aware analysis of weights while preserving orthonormality and allowing exact inverse reconstruction.

Subband-Wise Pruning. Let $\mathbf{W} \in \mathbb{R}^{m\times n}$ be a Transformer weight matrix with total element count $N = mn$, and let $\kappa \in (0, 1]$ denote the retention ratio in the wavelet domain. Correspondingly, the sparsity ratio is $1-\kappa$. After applying the 2D Haar wavelet transform, $\mathbf{W}$ is decomposed into four equally sized subbands: $\mathbf{LL}, \mathbf{LH}, \mathbf{HL}, \mathbf{HH} \in \mathbb{R}^{\frac{m}{2}\times\frac{n}{2}}$, corresponding to approximation and detail coefficients. We perform pruning directly in the wavelet domain by applying the same retention ratio κ to each subband independently. Specifically, let $\mathcal{S} = \{\mathrm{LL}, \mathrm{LH}, \mathrm{HL}, \mathrm{HH}\}$ denote the set of subband labels, let L_s denote the number of coefficients in each subband $s \in \mathcal{S}$, and note that $L_s = N/4$. We define the number of retained coefficients in each subband as:

$$k_s = \lfloor \kappa \cdot L_s \rfloor. \quad (2)$$

For each subband s, we retain the top-k_s coefficients ranked by absolute magnitude, setting all others to zero. This enforces a uniform retention ratio κ across frequency components. This subband-wise equal-ratio pruning strategy provides two key benefits. First, it avoids the disproportionate retention of low-frequency $\mathbf{LL}$ coefficients, which often dominate in magnitude due to energy concentration but offer limited spatial discriminability. Second, it mitigates over-pruning in high-frequency subbands ($\mathbf{LH}, \mathbf{HL}, \mathbf{HH}$), which encode localized and structural information critical to model expressiveness and stability. Unlike global magnitude pruning, which selects weights based solely on magnitude across all subbands, our method prunes each subband independently, preserving their structural balance. As a result, the pruned weights retain both low-frequency semantic context and high-frequency detail that is essential for maintaining performance in large-scale Transformer models.

2.2 Weight Reconstruction

Let $\widetilde{\mathbf{LL}}, \widetilde{\mathbf{LH}}, \widetilde{\mathbf{HL}}, \widetilde{\mathbf{HH}}$ denote the pruned subbands. The compressed weight matrix is reconstructed via the inverse Haar transform $\mathrm{iHWT}(\cdot)$:

$$\widetilde{\mathbf{W}} = \mathrm{iHWT}([\widetilde{\mathbf{LL}}, (\widetilde{\mathbf{LH}}, \widetilde{\mathbf{HL}}, \widetilde{\mathbf{HH}})]). \quad (3)$$

The result $\widetilde{\mathbf{W}} \in \mathbb{R}^{m\times n}$ retains the original shape. Since $\mathrm{HWT}(\cdot)$ is an orthogonal transform, the Frobenius norm of the reconstruction error between $\mathbf{W}$ and $\widetilde{\mathbf{W}}$ equals the corresponding coefficient error in the wavelet domain. In the current implementation, sparsity is imposed in the Haar wavelet coefficient domain rather than directly in the original weight domain. For deployment, the sparse subband coefficients are stored and the dense weight matrix is reconstructed once by inverse Haar transform before inference. Therefore, HWSP should be viewed as a calibration-free transform-domain compression method.

2.3 Layer-Wise Application to Transformers

We apply the proposed subband-wise pruning scheme to all linear projection matrices within Transformer blocks, including those in the self-attention mechanism (query, key, value, and output projections) as well as in the feedforward sublayers. For each weight matrix $\mathbf{W}$, the Haar transform, pruning, and reconstruction are performed independently. Although the global sparsity ratio $1 - \kappa$ is fixed across the model, we apply the corresponding retention ratio κ independently to each of the four Haar subbands of every matrix, ensuring balanced sparsification across frequency components. This design promotes modularity and enables highly parallel execution, as pruning decisions can be made independently for each layer without requiring inter-layer coordination or global optimization. Additionally, since the wavelet-based sparsification operates directly on the raw weight parameters, it naturally adapts to the local frequency characteristics of each layer, thereby balancing compression with the preservation of semantic content. The resulting subband-balanced sparsity pattern exhibits the following key properties:

- Uniform sparsification across subbands prevents the **LL** band from dominating the retained parameters, avoiding the bias toward energy-concentrated low-frequency components that commonly occurs in global magnitude pruning.
- By pruning each subband independently, the method retains localized, high-frequency information in the **LH**, **HL**, and **HH** bands, which is critical for maintaining the model's spatial expressiveness.
- The entire pruning procedure requires no training data, calibration samples, or activation statistics, making it particularly suitable for post-training deployment in resource-constrained settings.

2.4 Theoretical Properties

We present a set of theoretical results that establish the correctness and key structural properties of our wavelet-based pruning framework. First, we show that the two-dimensional Haar wavelet transform is both orthogonal and invertible, ensuring that reconstruction is exact in the absence of truncation. This guarantees that any approximation error arises solely from the pruning operation rather than from the transform itself. Formal proofs of all theorems are provided in the Appendix B.

Theorem 1 (Orthogonality and Exact Reconstruction of the Haar Transform). *Let* HWT *and* iHWT *denote the two-dimensional Haar wavelet transform and its inverse, respectively. For any input matrix $\mathbf{X} \in \mathbb{R}^{H \times W}$, the transform is perfectly invertible when no coefficient truncation is applied:*

$$\mathrm{iHWT}(\mathrm{HWT}(\mathbf{X})) = \mathbf{X}. \tag{4}$$

We next analyze the error introduced by coefficient truncation in the wavelet domain. In practice, pruning retains the k coefficients with the largest magnitudes. The following result gives an exact expression for the truncation error in coefficient space and shows that the corresponding reconstruction error is preserved in the original signal space under the Frobenius norm. Here, $\mathbf{w}$ denotes the vectorized form of the full Haar coefficient set, and $\mathrm{iHWT}(\mathbf{w})$ denotes reconstruction after reshaping $\mathbf{w}$ back to the corresponding wavelet-domain coefficient structure.

Theorem 2 (Top-k Truncation Error in Vector Form). *Let $\mathbf{w} \in \mathbb{R}^L$ be the full set of Haar wavelet coefficients, with their magnitudes sorted in descending order as $|w_{(1)}| \geq |w_{(2)}| \geq \cdots \geq |w_{(L)}|$. Let $\widetilde{\mathbf{w}} = T_k(\mathbf{w})$ denote the vector obtained by retaining the k coefficients with the largest magnitudes from $\mathbf{w}$ and setting the remaining $L - k$ coefficients to zero. Then the ℓ_2 truncation error in the coefficient space is given by:*

$$\|\mathbf{w} - \widetilde{\mathbf{w}}\|_2^2 = \sum_{i=k+1}^{L} |w_{(i)}|^2. \tag{5}$$

Moreover, if $\mathbf{X} = \mathrm{iHWT}(\mathbf{w})$ is the original signal and $\widetilde{\mathbf{X}} = \mathrm{iHWT}(\widetilde{\mathbf{w}})$ is the reconstructed signal from the truncated coefficients, then the reconstruction error in the signal space is:

$$\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F = \|\mathbf{w} - \widetilde{\mathbf{w}}\|_2. \tag{6}$$

Finally, we formalize a deterministic retention property of HWSP that follows directly from its equal-ratio subband design. Since HWSP applies the same retention ratio independently to each Haar subband, it guarantees a fixed minimum retained count for every subband. This distinguishes HWSP from global magnitude pruning, which operates on a joint cross-subband ranking and does not generally provide a prescribed per-subband retention guarantee under the same total budget.

Proposition 3 (Deterministic Per-Subband Retention Guarantee of HWSP). *Let* $\mathbf{W}$ *be a weight matrix and let* $\mathbf{w} = \mathrm{HWT}(\mathbf{W})$ *denote its two-dimensional Haar coefficients, partitioned into subbands* $S = \{\mathrm{LL}, \mathrm{LH}, \mathrm{HL}, \mathrm{HH}\}$. *Under* HWSP *with retention ratio* $\kappa \in (0, 1]$, *the number of retained coefficients in each subband* $s \in S$ *is*

$$k_s^{\mathrm{HWSP}} = \lfloor \kappa L_s \rfloor, \tag{7}$$

where L_s *is the number of coefficients in subband* s. *Therefore, for any designated subband* $s_0 \in S$ *and any target quota* $k_{s_0}^*$ *satisfying*

$$k_{s_0}^* \leq \lfloor \kappa L_{s_0} \rfloor, \tag{8}$$

HWSP *satisfies the quota deterministically:*

$$k_{s_0}^{\mathrm{HWSP}} \geq k_{s_0}^*. \tag{9}$$

Let the total retention budget of HWSP *be*

$$K^{\mathrm{HWSP}} = \sum_{s \in S} \lfloor \kappa L_s \rfloor. \tag{10}$$

By contrast, global magnitude pruning under the same total retention budget K^{HWSP} *does not, in general, provide a deterministic per-subband quota guarantee.*

3 Experiments

3.1 Experimental Setup

Models, Datasets & Evaluation. **(i)** We evaluate our method on a broad set of LLMs, including LLaMA-7B, LLaMA2-7B, LLaMA3-8B, LLaMA3-70B (Ma et al., 2023), OPT-1.3B, OPT-6.7B, OPT-13B (Zhang et al., 2022), Qwen2.5-7B (Yang et al., 2024b) and Qwen3-8B (Yang et al., 2025a). These cover both the LLaMA, OPT and Qwen families across different model scales, offering a comprehensive evaluation of pruning performance. For zero-shot evaluation, we adopt seven widely used benchmark datasets: OpenBookQA (Mihaylov et al., 2018), ARC-Easy, ARC-Challenge (Clark et al., 2018), Winogrande (Sakaguchi et al., 2021), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), and MathQA (Amini et al., 2019). All zero-shot evaluations are conducted using the LM-Evaluation-Harness (Gao et al., 2024) framework. These tasks span commonsense reasoning, science QA, and mathematical inference, providing diverse evaluation challenges. For generative evaluation, we use WikiText2 (Merity et al., 2016), C4 (Raffel et al., 2020), and PTB_Text_Only (Marcus et al., 1993), reporting word-level perplexity (PPL) as the primary metric. No calibration data or post-pruning weight updates are employed throughout all experiments. **(ii)** Given that our method does not rely on calibration data or weight updates, we further evaluate its effectiveness on multimodal models. Specifically, we evaluate HWSP on Janus-Pro-1B and Janus-Pro-7B (Chen et al., 2025) using three benchmarks: POPE (Li et al., 2023), MMB (Liu et al., 2024b), and MM-VET (Yu et al., 2023). These cover visual grounding, factual reasoning, and compositional understanding, extending our analysis to vision-language models under sparsity. To ensure a fair comparison, all evaluations are conducted using the VLMEvalKit (Duan et al., 2024) framework.

Baselines. We compare HWSP against several representative post-training pruning methods that vary in their reliance on calibration data and weight updates. Magnitude Pruning (Zhu and Gupta, 2017) serves as a simple and strong baseline. It prunes weights based solely on their absolute magnitudes without requiring any calibration data or parameter updates. SparseGPT (Frantar and Alistarh, 2023) prunes via a second-order strategy that solves a layer-wise reconstruction problem using Hessian approximations. Although it achieves strong accuracy, it requires calibration data and incurs high computational cost from per-layer weight updates. SparseLLM (Bai et al., 2024) extends global pruning to higher sparsity by introducing auxiliary variables that decompose the optimization into parallelizable subproblems. However, this design also leads to substantial computational overhead. Wanda (Sun et al., 2024) adopts a lightweight strategy based on the product of weight magnitudes and activation norms to estimate importance scores. It avoids Hessian computations but still requires calibration samples. To ensure a fair comparison, we adopt the evaluation protocol from SparseGPT and Wanda by pruning only the linear projection layers in Transformer blocks. We use magnitude pruning as the primary baseline because it shares the same calibration-free and training-free setting as HWSP. This setup allows us to directly compare the effectiveness of HWSP against conventional magnitude-based methods under identical conditions.

Implementation Details. To evaluate the proposed HWSP framework, we follow the standard post-

		LLaMA-7B			LLaMA2-7B			LLaMA3-8B		
Sparsity	*Method*	**WikiText2**	**C4**	**PTB**	**WikiText2**	**C4**	**PTB**	**WikiText2**	**C4**	**PTB**
0%	Dense	5.67	7.33	8.34	5.47	7.28	24.09	6.23	9.57	9.88
20%	SparseGPT	5.79	7.47	8.50	5.60	7.48	27.62	**6.38**	10.00	**9.93**
	SparseLLM	–	–	–	5.61	7.48	27.52	6.40	9.96	10.01
	Wanda	5.81	7.46	8.51	5.60	7.45	27.53	6.42	**9.94**	10.01
	Magnitude	6.02	7.76	8.81	5.71	7.63	32.38	6.84	10.52	10.41
	HWSP	**5.74**	**7.42**	**8.45**	**5.56**	**7.40**	**25.08**	6.46	9.97	10.06
40%	SparseGPT	**6.18**	8.33	9.20	**5.94**	8.31	35.24	**7.17**	**11.96**	**11.03**
	SparseLLM	–	–	–	6.25	8.46	**29.50**	7.27	12.20	11.16
	Wanda	6.32	**8.22**	**9.14**	6.05	**8.17**	29.91	7.40	11.97	**11.03**
	Magnitude	8.60	11.50	13.41	7.92	10.78	169.28	19.64	26.29	21.55
	HWSP	6.46	8.37	9.47	6.32	8.39	31.43	8.65	13.38	12.63

Table 1: Perplexity (↓) of LLaMA families under 20% and 40% sparsity. The dash indicates a pruning failure caused by a non–positive-definite matrix in Cholesky decomposition. The average pruning times on a single A100 GPU are: SparseGPT (8 min), SparseLLM (130 min), Wanda (2 min), Magnitude (1 s), and HWSP (12 s).

training pruning paradigm without retraining. All experiments are conducted in a calibration-free and training-free setting. We prune weights layer-wise across all linear projection matrices in Transformer blocks and apply uniform subband-wise truncation in the Haar wavelet domain. All evaluations are conducted on NVIDIA A100 GPUs with 80GB memory. All experiments are conducted with HuggingFace Transformers (Wolf et al., 2020), using consistent seeds to ensure reproducibility.

3.2 Language Modeling Results

Table 1 reports the perplexity (PPL) of pruned models on WikiText2, C4, and PTB under sparsity levels of 20% and 40%. At 20% sparsity, HWSP achieves competitive performance compared with SparseGPT, SparseLLM, and Wanda across most evaluated models and datasets. For LLaMA-7B, HWSP achieves a PPL of 5.74 on WikiText2 and 8.45 on PTB, indicating stronger language modeling performance than methods that require calibration data and weight updates. Similarly, on LLaMA2-7B, HWSP achieves the lowest perplexity on WikiText2, C4, and PTB, obtaining scores of 5.56, 7.40, and 25.08, respectively. These results demonstrate superior robustness under moderate compression. On LLaMA3-8B, the method maintains strong performance across all datasets, confirming its generalization across model scales. At 40% sparsity, HWSP demonstrates clear advantages over magnitude pruning. On LLaMA-7B and LLaMA2-7B, it consistently achieves lower PPL, reflecting stronger preservation of generative capability under aggressive sparsity. For example, on the PTB dataset, HWSP reduces PPL from 13.41 to 9.47 on LLaMA-7B, and from 169.28 to 31.43 on LLaMA2-7B. On LLaMA3-8B, HWSP consistently outperforms magnitude pruning in terms of perplexity across all evaluated datasets, achieving lower PPL on WikiText2, PTB, and C4. In addition, compared to pruning methods that require calibration data and weight updates, HWSP attains comparable levels of PPL, while operating with significantly lower computational overhead. Detailed results for OPT and Qwen models are provided in Appendix C.

3.3 Zero-shot Task Evaluation

We evaluate the generalization capability of HWSP on a diverse set of zero-shot tasks, covering commonsense reasoning, scientific question answering, and mathematical problem solving. Experiments are conducted on LLaMA3-8B and OPT-6.7B across seven widely used benchmarks: OpenBookQA, ARC-Easy, ARC-Challenge, Winogrande, HellaSwag, PIQA, and MathQA, as summarized in Table 2. HWSP achieves competitive average accuracy at 20% sparsity despite operating without calibration data or weight updates. Compared with magnitude pruning, HWSP generally achieves better average accuracy under the same sparsity constraints, with clearer improvements at 40% sparsity. These results highlight the effectiveness of our method in preserving downstream task performance under post-training constraints.

		Method	**Openb.**	**ARC_e**	**ARC_c**	**WinoG.**	**HellaS.**	**PIQA**	**MathQA**	**Average↑**
LLaMA3-8B	0%	Dense	33.40	81.40	51.37	73.72	60.02	80.03	39.73	59.95
	20%	SparseGPT	34.20	80.77	50.26	74.19	59.98	79.76	39.03	59.74
		SparseLLM	33.80	80.60	49.74	73.72	60.06	79.76	39.26	59.71
		Wanda	33.00	80.89	50.43	73.80	60.03	79.87	39.97	59.71
		Magnitude	33.60	80.89	49.91	73.95	59.65	78.89	39.26	59.45
		HWSP	33.80	81.65	51.19	74.43	59.76	79.22	38.96	**59.86**
	40%	SparseGPT	31.40	78.45	47.44	72.77	56.74	77.91	35.98	57.24
		SparseLLM	29.60	78.03	46.33	74.03	56.45	77.64	35.64	56.82
		Wanda	30.80	78.41	48.46	73.16	56.85	77.91	36.48	**57.44**
		Magnitude	27.20	67.59	40.10	65.35	47.18	72.74	32.86	50.43
		HWSP	31.00	74.33	42.92	70.56	55.69	78.29	35.54	55.48
OPT-6.7B	0%	Dense	27.00	65.57	30.38	64.40	50.44	76.44	24.46	48.38
	20%	SparseGPT	27.60	65.45	30.20	64.48	50.20	76.22	24.56	48.39
		SparseLLM	27.60	65.03	30.89	65.19	50.34	76.12	25.03	**48.60**
		Wanda	28.40	65.49	30.97	64.88	50.13	76.50	24.66	48.58
		Magnitude	26.00	65.36	30.72	64.01	49.94	76.39	25.59	48.28
		HWSP	26.00	64.73	30.03	65.35	49.85	76.28	24.52	48.11
	40%	SparseGPT	27.40	65.19	29.78	64.25	48.69	74.92	24.32	47.79
		SparseLLM	28.00	65.11	29.52	64.64	48.73	75.24	24.49	**47.96**
		Wanda	27.20	65.07	29.35	62.59	47.58	74.86	25.13	47.40
		Magnitude	24.00	61.53	27.05	58.25	46.30	73.94	23.58	44.95
		HWSP	24.60	62.37	29.78	63.54	45.84	74.37	24.66	46.45

Table 2: Accuracy (↑) of pruning methods on LLaMA3-8B and OPT-6.7B by lm_eval v0.4.7.

Method	**WikiText2**	**PTB**	**C4**	**Time (min)**	**Speedup**	**A100 GPUs**	**Max GPU Mem (MiB)**
Dense	7.05	27.83	22.23	–	–	–	–
SparseGPT	7.79	28.44	23.03	84	1.0×	3	52,566
SparseLLM	–	–	–	–	–	–	–
Wanda	7.77	28.33	22.76	8	10.5×	3	54,828
Magnitude	9.11	31.23	25.34	2	42.0×	1	18,021
HWSP	7.88	28.05	22.94	3	28.0×	1	15,689

Table 3: Perplexity, pruning time, speedup, and memory usage on **LLaMA3-70B** at 20% sparsity. Speedup is relative to SparseGPT, and memory usage reflects average peak consumption per A100 GPU. ***Note.*** We do not report SparseLLM results on LLaMA3-70B because its official codebase does not currently support models of this scale.

3.4 Pruning Speed on LLaMA3-70B

We further assess the computational efficiency of HWSP on a large-scale language model by conducting experiments on LLaMA3-70B at 20% sparsity. We compare HWSP with representative pruning baselines in terms of pruning time, GPU usage, peak memory consumption, and perplexity on WikiText2, PTB, and C4, as summarized in Table 3. HWSP completes the pruning process in only 3 minutes using a single A100 GPU, yielding a 28× speedup over SparseGPT, which takes 84 minutes across three A100 GPUs. In terms of memory usage, HWSP requires only 15.6 GB at peak, which is less than one-third of the peak memory usage of both Wanda and SparseGPT. Despite its lightweight resource requirements, HWSP achieves perplexity closely comparable to those of SparseGPT and Wanda across all three benchmarks. These results demonstrate that HWSP scales effectively to extremely large models while maintaining a favorable balance between pruning efficiency and model quality.

3.5 Applying HWSP to Multimodal Janus-Pro Models

Given that HWSP does not rely on calibration data or parameter updates, it naturally extends to large-scale multimodal vision–language models

		Janus-Pro-1B			Janus-Pro-7B		
Sparsity	*Method*	**POPE**	**MMB**	**MM-VET**	**POPE**	**MMB**	**MM-VET**
0%	Dense	86.2	75.5	39.8	87.4	79.2	50.0
20%	Magnitude	82.3	45.7	31.2	**77.2**	**65.3**	44.1
	HWSP	**84.4**	**57.3**	**34.7**	75.2	**65.3**	**47.1**
40%	Magnitude	NaN	NaN	NaN	53.4	50.7	29.9
	HWSP	80.4	14.2	21.8	**61.9**	**60.9**	**38.5**

Table 4: Performance (%) of Janus-Pro families at varying sparsity ratios on multimodal understanding benchmarks. NaN indicates invalid model outputs.

		LLaMA-7B			LLaMA2-7B			LLaMA3-8B		
Sparsity	*Method*	**WikiText2**	**C4**	**PTB**	**WikiText2**	**C4**	**PTB**	**WikiText2**	**C4**	**PTB**
0%	Dense	5.67	7.33	8.34	5.47	7.28	24.09	6.23	9.57	9.88
80%	SparseGPT	65.70	2e2	1e2	53.52	1e2	2e3	68.75	2e2	2e2
	SparseLLM	–	–	–	54.00	1e2	1e3	68.80	2e2	2e2
	Wanda	3e3	2e3	5e3	2e3	2e3	3e3	1e3	1e3	2e3
	Magnitude	1e5	1e5	7e4	1e5	1e5	5e4	1e8	5e7	1e7
	HWSP	5e4	3e4	4e4	2e4	3e4	1e4	2e6	1e5	1e5

Table 5: Perplexity (↓) on WikiText2, C4, and PTB datasets for pruned LLaMA-7B, LLaMA2-7B, and LLaMA3-8B models under 80% sparsity.

without modification. We evaluate this capability on two representative models, Janus-Pro-1B and Janus-Pro-7B, using three standard benchmarks: POPE, MMB, and MM-VET, which cover multimodal reasoning tasks such as visual grounding and compositional inference. As shown in Table 4, HWSP often outperforms magnitude pruning under the same sparsity levels, with clearer gains on several multimodal benchmarks. At 20% sparsity, it improves MM-VET accuracy by 3.5 points on Janus-Pro-1B and 3.0 points on Janus-Pro-7B. At 40%, magnitude pruning fails on Janus-Pro-1B, while HWSP remains functional and achieves up to 80.4% on POPE. For Janus-Pro-7B at 40% sparsity, HWSP improves performance on all three benchmarks, with gains of 8.5, 10.2, and 8.6 percentage points on POPE, MMB, and MM-VET, respectively. These results highlight the robustness and generalizability of HWSP in multimodal settings under aggressive compression.

4 Conclusion

We presented HWSP, a post-training pruning framework that avoids both calibration data and weight updates. By applying a two-dimensional Haar transform and uniformly pruning each frequency subband, HWSP preserves both low- and high-frequency information in a balanced manner. Theoretical analysis clarifies that HWSP provides a deterministic per-subband retention guarantee under its equal-ratio design. Experiments on LLaMA, OPT and Qwen models show that HWSP achieves competitive accuracy relative to strong baselines while significantly improving pruning efficiency. Compared with magnitude pruning, it generally delivers better performance across many sparsity levels and model sizes. Our empirical results demonstrate that balanced subband pruning is more effective than over-preserving only dominant low-frequency components, reinforcing the value of balanced preservation across frequency components for maintaining model accuracy.

Limitation

Table 5 reports the perplexity results on WikiText2, C4, and PTB for LLaMA-7B, LLaMA2-7B, and LLaMA3-8B under 80% sparsity. Although HWSP consistently achieves lower perplexity than magnitude pruning across all evaluated models and datasets at this sparsity level, the gap between the pruned models and their dense counterparts remains substantial. This suggests that, under extreme sparsity, static pruning alone is still insufficient to fully preserve model quality. Moreover, data-calibrated methods such as Wanda also exhibit clear performance degradation in this regime.

While parameter-update-based methods, such as those using second-order approximations, may further improve performance under extreme sparsity, they typically introduce additional computational cost.

Another limitation lies in the current deployment form of HWSP. Specifically, sparsity is imposed in the Haar wavelet coefficient domain rather than directly in the original weight domain. In the current setting, the sparse subband coefficients are stored, and the dense weight matrix is reconstructed once by inverse Haar transform before inference. As a result, HWSP mainly provides storage-side compression, but does not directly inherit the sparse-kernel acceleration benefits typically associated with standard weight-domain unstructured pruning. In future work, we plan to extend HWSP toward more deployment-friendly structured pruning or operator-level designs, so that wavelet-based subband allocation can better support efficient inference.

Acknowledgments

This work was funded by National Natural Science Foundation of China (Grant No. 62366036, 62406321), Outstanding Youth Fund Project of Inner Mongolia Autonomous Region (Grant No. 2025JQ010), Program for Young Talents of Science and Technology in Universities of Inner Mongolia Autonomous Region (Grant No. NJYT24033), Major Science and Technology Projects of Inner Mongolia Autonomous Region (Grant No. 2025ZDSF0029), Key R&D and Achievement Transformation Program of Inner Mongolia Autonomous Region (Grant No. 2025YFDZ0011, 2025YFDZ0026, 2025YFSH0021, 2025YFHH0073), Hohhot Science and Technology Project (Grant No. 2023-Zhan-Zhong-1).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

Mohammad Babaeizadeh, Paris Smaragdis, and Roy H Campbell. 2016. Noiseout: A simple way to prune neural networks. *arXiv preprint arXiv:1611.06211*.

Guangji Bai, Yijiang Li, Chen Ling, Kibaek Kim, and Liang Zhao. 2024. Sparsellm: Towards global pruning for pre-trained language models. *arXiv preprint arXiv:2402.17946*.

Hritik Bansal, Karthik Gopalakrishnan, Saket Dingliwal, Sravan Bodapati, Katrin Kirchhoff, and Dan Roth. 2023. Rethinking the role of scale for in-context learning: An interpretability-based case study at 66 billion scale. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11833–11856, Toronto, Canada. Association for Computational Linguistics.

Samiul Basir Bhuiyan, Md Sazzad Hossain Adib, Mohammed Aman Bhuiyan, Muhammad Rafsan Kabir, Moshiur Farazi, Shafin Rahman, and Nabeel Mohammed. 2025. Z-prunear: Post-training pruning of large language models for efficiency without retraining. In *2025 IEEE/ACS 22nd International Conference on Computer Systems and Applications (AICCSA)*, pages 1–8. IEEE.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. 2025. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *Preprint*, arXiv:2501.17811.

Xinrui Chen, Hongxing Zhang, Fanyi Zeng, Yongxian Wei, Yizhi Wang, Xitong Ling, Guanghao Li, and Chun Yuan. 2026. Prune&comp: Free lunch for layer-pruned llms via iterative pruning with magnitude compensation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 20316–20324.

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*.

Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, and 1 others. 2024. Vlmevalkit: An open-source toolkit for evaluating large multi-modality models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 11198–11201.

Abhimanyu Dubey, Moitreya Chatterjee, and Narendra Ahuja. 2018. Coreset-based neural network compression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 454–470.

Moshe Eliasof, Benjamin J Bodner, and Eran Treister. 2023. Haar wavelet feature compression for quantized graph convolutional networks. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4):4542–4553.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. 2023. Depgraph: Towards any structural pruning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16091–16101.

Elias Frantar and Dan Alistarh. 2022. Spdy: Accurate pruning with speedup guarantees. In *International conference on machine learning*, pages 6726–6743. PMLR.

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.

Advait Harshal Gadhikar, Sohom Mukherjee, and Rebekka Burkholz. 2023. Why random pruning is all we need to start sparse. In *International Conference on Machine Learning*, pages 10542–10570. PMLR.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. A framework for few-shot language model evaluation.

Jialong Guo, Xinghao Chen, Yehui Tang, and Yunhe Wang. 2025. Slimllm: Accurate structured pruning for large language models. In *International Conference on Machine Learning*, pages 20766–20776. PMLR.

Song Han, Huizi Mao, and William J Dally. 2015a. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

Song Han, Jeff Pool, John Tran, and William Dally. 2015b. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28.

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

Jiujun He and Huazhen Lin. 2025. Olica: Efficient structured pruning of large language models without retraining. In *International Conference on Machine Learning*, pages 22580–22594. PMLR.

Duc NM Hoang and Shiwei Liu. 2023. Revisiting pruning at initialization through the lens of ramanujan graph. *ICLR 2023*.

Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. 2021. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *Journal of Machine Learning Research*, 22(241):1–124.

Bairu Hou, Qibin Chen, Jianyu Wang, Guoli Yin, Chong Wang, Nan Du, Ruoming Pang, Shiyu Chang, and Tao Lei. 2025. Instruction-following pruning for large language models. In *International Conference on Machine Learning*, pages 23894–23909. PMLR.

Hanyu Hu, Pengxiang Zhao, Ping Li, Yi Zheng, Zhefeng Wang, and Xiaoming Yuan. 2025. Fasp: Fast and accurate structured pruning of large language models. *arXiv preprint arXiv:2501.09412*.

Hengyuan Hu, Rui Peng, Yu-Wing Tai, and Chi-Keung Tang. 2016. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv preprint arXiv:1607.03250*.

Itay Hubara, Brian Chmiel, Moshe Island, Ron Banner, Joseph Naor, and Daniel Soudry. 2021. Accelerated sparse neural training: A provable and efficient method to find n: m transposable masks. *Advances in neural information processing systems*, 34:21099–21111.

Yixin Ji, Yang Xiang, Juntao Li, Qingrong Xia, Ping Li, Xinyu Duan, Zhefeng Wang, and Min Zhang. 2025. Beware of calibration data for pruning large language models. In *The Thirteenth International Conference on Learning Representations*.

Aditya Kusupati, Vivek Ramanujan, Raghav Somani, Mitchell Wortsman, Prateek Jain, Sham Kakade, and Ali Farhadi. 2020. Soft threshold weight reparameterization for learnable sparsity. In *International Conference on Machine Learning*, pages 5544–5555. PMLR.

Woosuk Kwon, Sehoon Kim, Michael W Mahoney, Joseph Hassoun, Kurt Keutzer, and Amir Gholami. 2022. A fast post-training pruning framework for transformers. *Advances in Neural Information Processing Systems*, 35:24101–24116.

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

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Evaluating object hallucination in large vision-language models. *arXiv preprint arXiv:2305.10355*.

Shiwei Liu, Tianlong Chen, Zhenyu Zhang, Xuxi Chen, Tianjin Huang, Ajay Jaiswal, and Zhangyang Wang. 2023a. Sparsity may cry: Let us fail (current) sparse neural networks together! *arXiv preprint arXiv:2303.02141*.

Yajun Liu, Kefeng Fan, and Wenju Zhou. 2024a. Fpwt: Filter pruning via wavelet transform for cnns. *Neural Networks*, 179:106577.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, and 1 others. 2024b. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pages 216–233. Springer.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. 2017. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pages 2736–2744.

Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. 2018. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*.

Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, and 1 others. 2023b. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*, pages 22137–22176. PMLR.

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*.

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

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2016. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*.

Rui Pan, Shivanshu Shekhar, Boyao Wang, Shizhe Diao, Jipeng Zhang, Xingyuan Pan, Renjie Pi, and Tong Zhang. 2025. Adapt-pruner: Adaptive structural pruning for efficient small language model training. *arXiv preprint arXiv:2502.03460*.

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.

Alex Renda, Jonathan Frankle, and Michael Carbin. 2020. Comparing rewinding and fine-tuning in neural network pruning. *arXiv preprint arXiv:2003.02389*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Victor Sanh, Thomas Wolf, and Alexander Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in neural information processing systems*, 33:20378–20389.

Maying Shen, Hongxu Yin, Pavlo Molchanov, Lei Mao, Jianna Liu, and Jose M Alvarez. 2022. Structural pruning via latency-saliency knapsack. *Advances in Neural Information Processing Systems*, 35:12894–12908.

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

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*.

Prateek Verma. 2024. Waveletgpt: Wavelets meet large language models. *arXiv preprint arXiv:2409.12924*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and 1 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models. *arXiv preprint arXiv:2204.00408*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024a. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, and 1 others. 2024b. Qwen2.5 technical report. *ArXiv*, abs/2412.15115.

Yifan Yang, Kai Zhen, Bhavana Ganesh, Aram Galstyan, Goeric Huybrechts, Markus Müller, Jonas M Kübler, Rupak Vignesh Swaminathan, Athanasios Mouchtaris, Sravan Babu Bodapati, and 1 others. 2025b. Wanda++: Pruning large language models via regional gradients. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 4321–4333.

Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. 2023. Mm-vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*.

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*.

Mingjin Zhang, Handi Yang, Jie Guo, Yunsong Li, Xinbo Gao, and Jing Zhang. 2024. Irprunedet: efficient infrared small target detection via wavelet structure-regularized soft channel pruning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 7224–7232.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and 1 others. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Pu Zhao, Fei Sun, Xuan Shen, Pinrui Yu, Zhenglun Kong, Yanzhi Wang, and Xue Lin. 2024. Pruning foundation models for high accuracy without retraining. *arXiv preprint arXiv:2410.15567*.

Yefan Zhou, Yaoqing Yang, Arin Chang, and Michael W Mahoney. 2023. A three-regime model of network pruning. In *International Conference on Machine Learning*, pages 42790–42809. PMLR.

Michael Zhu and Suyog Gupta. 2017. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*.

Appendix

A Related work

Pruning Methods. Network pruning (LeCun et al., 1989; Hassibi et al., 1993) is a widely used technique for compressing deep neural networks by removing redundant weights, thereby reducing both memory usage and computational cost. Existing pruning methods are commonly divided into structured and unstructured approaches, depending on whether parameters are removed in regular groups or at the individual weight level. Structured pruning removes architectural units such as filters, channels, or attention heads, and is often favored for hardware-efficient acceleration (Liu et al., 2017; Molchanov et al., 2019; Fan et al., 2019; Shen et al., 2022; Xia et al., 2022; Fang et al., 2023). Some methods further rely on activation statistics, such as zero-activation ratios or mean activations, to guide pruning decisions (Hu et al., 2016; Molchanov et al., 2016; Babaeizadeh et al., 2016; Dubey et al., 2018). More recently, structured pruning has been extended to LLMs, where prior studies have revealed task- and prompt-dependent sparsity patterns in attention heads and feedforward neurons (Ma et al., 2023; Bansal et al., 2023; Liu et al., 2023b). Recent work has continued this line of research. Adapt-Pruner studies adaptive structural pruning with further training for efficient small language model construction, while Instruction-Following Pruning proposes an instruction-conditioned dynamic pruning strategy (Pan et al., 2025; Hou et al., 2025). Other structured methods such as FASP, SlimLLM, and Olica focus on improving pruning accuracy and efficiency under practical deployment constraints (Hu et al., 2025; Guo et al., 2025; He and Lin, 2025).

Unstructured pruning methods, in contrast, operate at the individual weight level and offer fine-grained compression (Han et al., 2015b,a; Hoang and Liu, 2023; Gadhikar et al., 2023; Liu et al., 2023a). In conventional settings, these methods often rely on retraining (Liu et al., 2018; Zhou et al., 2023), training-time modifications (Sanh et al., 2020; Kusupati et al., 2020), or iterative optimization (Renda et al., 2020). Such requirements are difficult to scale to modern LLMs because of their size and training cost (Zhang et al., 2022; Touvron et al., 2023). To avoid full retraining, recent post-training pruning methods use limited calibration data together with layer-wise reconstruction (Hubara et al., 2021; Frantar and Alistarh, 2022; Kwon et al., 2022). SparseGPT (Frantar and Alistarh, 2023) extends this idea to LLMs through second-order reconstruction, while Wanda (Sun et al., 2024) uses a lightweight importance score based on weight and activation statistics. Wanda++ (Yang et al., 2025b) further extends this line of work by incorporating decoder-block-level regional gradients and a regional optimization procedure to reduce the discrepancy between dense and sparse decoder

Method	Weight Update	Calibration Data	Pruning Metric S_{ij}	Complexity
SparseGPT	✓	✓	$\left[\lvert W\rvert^2/\mathrm{diag}\left(\left(\mathbf{X}\mathbf{X}^\top+\lambda\mathbf{I}\right)^{-1}\right)\right]_{ij}$	$\mathcal{O}(d^3_{\text{hidden}})$
SparseLLM	✓	✓	$(W_{ij}\cdot\bar{a}_j)^2$	$\mathcal{O}(d^3_{\text{hidden}})$
Wanda	✗	✓	$\lvert W_{ij}\rvert\cdot\lVert X_j\rVert_2$	$\mathcal{O}(d^2_{\text{hidden}})$
Magnitude	✗	✗	$\lvert W_{ij}\rvert$	$\mathcal{O}(1)$
HWSP (ours)	✗	✗	$\lvert W_{ij}\rvert$	$\mathcal{O}(d_{\text{hidden}})$

Table 6: Comparison of HWSP with existing pruning algorithms for LLMs. Note: The pruning metric for SparseLLM is derived from its least-squares optimization objective. $(W_{ij}\cdot\bar{a}_j)^2$ approximates the contribution of each weight to the reconstruction error of pre-activation outputs z_ℓ, where $\bar{a}_j$ is the average activation over calibration samples.

outputs. SparseLLM (Bai et al., 2024) further improves pruning under higher sparsity by introducing auxiliary variables that decompose the optimization into parallelizable subproblems, although this also increases computational overhead. Z-Pruner explored retraining-free post-training pruning for LLMs, and Prune&Comp studied compensation for training-free layer pruning (Bhuiyan et al., 2025; Chen et al., 2026). Despite this progress, most competitive methods still rely on calibration data, weight updates, or additional compensation steps. Magnitude pruning remains attractive because of its simplicity and low cost, but it typically suffers substantial quality degradation once sparsity becomes moderately high (Frantar and Alistarh, 2023). A comparison of representative LLM pruning methods is shown in Table 6. Unlike most existing approaches, our method requires neither calibration data nor parameter updates, while maintaining complexity and runtime close to those of magnitude pruning.

Wavelet Transform. Recent studies have explored wavelet transforms as a tool for improving neural network efficiency. In the CNN domain, Liu et al. (Liu et al., 2024a) proposed FPWT (Filter Pruning via Wavelet Transform), which applies the Haar transform to activation feature maps and prunes filters according to cosine similarity in the frequency domain. Wu et al. (Eliasof et al., 2023) designed a Haar-based compression method for quantized GCNs by reducing redundancy in graph embeddings. For infrared small object detection, IRPruneDet (Zhang et al., 2024) incorporates wavelet regularization into channel pruning to preserve semantic features and improve detection sensitivity. Wavelet-based ideas have also appeared in LLM architecture design. WaveletGPT (Verma, 2024), for example, introduces orthogonal wavelet bases into the token embedding stage to improve structural modeling without adding parameters. However, these methods do not address post-training pruning of LLM weights. FPWT operates on activation feature maps in CNNs and requires forward passes together with similarity analysis, which makes it difficult to transfer directly to LLM pruning. By contrast, our method applies Haar wavelet decomposition directly to LLM weight matrices and performs pruning in the wavelet domain without calibration data or weight updates. This setting is lightweight, theoretically grounded, and well suited to post-training compression.

B Theoretical Proofs

Theorem 1 (Orthogonality and Exact Reconstruction of the Haar Transform). *Let* HWT *and* iHWT *denote the two-dimensional Haar wavelet transform and its inverse, respectively. For any input matrix* $\mathbf{X}\in\mathbb{R}^{H\times W}$*, the transform is perfectly invertible when no coefficient truncation is applied:*

$$\mathrm{iHWT}(\mathrm{HWT}(\mathbf{X}))=\mathbf{X}. \tag{11}$$

Proof. For a two-dimensional input matrix $\mathbf{X}\in\mathbb{R}^{H\times W}$, the 2D Haar transform can be written as a separable operation that applies orthonormal transforms along the two spatial dimensions. Let $H_H\in\mathbb{R}^{H\times H}$ and $H_W\in\mathbb{R}^{W\times W}$ denote the orthonormal Haar basis matrices along the row and column dimensions, respectively. Then the forward transform is

$$\mathrm{HWT}(\mathbf{X})=H_H\mathbf{X}H_W^\top, \tag{12}$$

and the inverse transform is

$$\mathrm{iHWT}(\mathbf{Y})=H_H^\top\mathbf{Y}H_W. \tag{13}$$

Since $H_H^\top H_H=I$ and $H_W^\top H_W=I$, we obtain

$$\mathrm{iHWT}(\mathrm{HWT}(\mathbf{X}))=H_H^\top H_H\mathbf{X}H_W^\top H_W=\mathbf{X}. \tag{14}$$

		OPT-1.3B			OPT-6.7B			OPT-13B		
Sparsity	*Method*	**WikiText2**	**C4**	**PTB**	**WikiText2**	**C4**	**PTB**	**WikiText2**	**C4**	**PTB**
0%	Dense	14.63	15.67	15.64	10.58	12.49	12.15	10.11	11.90	11.46
20%	SparseGPT	15.07	**15.88**	**15.95**	**10.85**	**12.56**	**12.25**	10.09	**11.97**	**11.48**
	SparseLLM	15.07	**15.88**	15.98	**10.85**	**12.56**	**12.25**	**10.08**	12.01	11.49
	Wanda	15.07	16.17	16.04	10.79	12.74	12.44	10.07	12.15	11.57
	Magnitude	15.61	16.36	16.32	11.26	13.01	12.76	10.62	12.54	11.99
	HWSP	**14.83**	15.95	16.01	10.92	12.66	12.39	10.21	12.08	11.64
40%	SparseGPT	25.09	21.50	24.18	11.01	13.41	**12.99**	**10.33**	**12.69**	**12.01**
	SparseLLM	25.39	21.33	24.22	**10.87**	**13.40**	**12.99**	10.43	12.70	12.22
	Wanda	**16.55**	**18.83**	**18.39**	11.11	14.95	14.38	10.66	13.88	13.06
	Magnitude	387.9	115.1	194.3	31.90	21.29	24.94	99.25	80.69	102.4
	HWSP	18.68	19.24	20.68	13.17	15.37	15.88	12.59	14.75	15.58

Table 7: Perplexity on WikiText2, C4, and PTB datasets for pruned OPT-1.3B, OPT-6.7B, and OPT-13B models under 20% and 40% sparsity.

Therefore, the 2D Haar transform is perfectly invertible when no coefficient truncation is applied. □

Discussion. This property shows that the transform itself introduces no approximation error before thresholding. It also justifies performing sparsification in the wavelet domain, because any loss comes entirely from coefficient removal rather than from the transform.

Theorem 2 (Top-k Truncation Error in Vector Form). *Let $\mathbf{w} \in \mathbb{R}^L$ be the full set of Haar wavelet coefficients, with magnitudes sorted in descending order as $|w_{(1)}| \geq |w_{(2)}| \geq \cdots \geq |w_{(L)}|$. Let $\widetilde{\mathbf{w}} = T_k(\mathbf{w})$ denote the vector obtained by retaining the k coefficients with the largest magnitudes from $\mathbf{w}$ and setting the remaining $L-k$ coefficients to zero. Then the ℓ_2 truncation error in coefficient space is*

$$\|\mathbf{w} - \widetilde{\mathbf{w}}\|_2^2 = \sum_{i=k+1}^{L} |w_{(i)}|^2. \tag{15}$$

Moreover, if $\mathbf{X} = \mathrm{iHWT}(\mathbf{w})$ is the original signal and $\widetilde{\mathbf{X}} = \mathrm{iHWT}(\widetilde{\mathbf{w}})$ is the reconstructed signal from the truncated coefficients, then the reconstruction error in signal space is

$$\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F = \|\mathbf{w} - \widetilde{\mathbf{w}}\|_2. \tag{16}$$

Proof. By the definition of $T_k(\mathbf{w})$, the truncated coefficient vector $\widetilde{\mathbf{w}}$ differs from $\mathbf{w}$ only in the $L-k$ positions corresponding to the coefficients with the smallest magnitudes. These discarded coefficients are precisely $|w_{(k+1)}|, |w_{(k+2)}|, \ldots, |w_{(L)}|$. Therefore, the squared ℓ_2 norm of the difference vector $(\mathbf{w} - \widetilde{\mathbf{w}})$ is exactly the sum of the squared magnitudes of the discarded coefficients:

$$\|\mathbf{w} - \widetilde{\mathbf{w}}\|_2^2 = \sum_{j \in \mathcal{D}} |w_j|^2 = \sum_{i=k+1}^{L} |w_{(i)}|^2, \tag{17}$$

where $\mathcal{D}$ denotes the index set of discarded coefficients. This proves Eq. (15).

Next, from the orthonormality of the Haar basis matrices used in Theorem 1, the inverse Haar transform preserves Euclidean energy in coefficient space and Frobenius energy in signal space. Applying this property to the difference vector $(\mathbf{w} - \widetilde{\mathbf{w}})$ gives

$$\|\mathrm{iHWT}(\mathbf{w} - \widetilde{\mathbf{w}})\|_F = \|\mathbf{w} - \widetilde{\mathbf{w}}\|_2. \tag{18}$$

Since iHWT is linear, we also have

$$\begin{aligned} \mathrm{iHWT}(\mathbf{w} - \widetilde{\mathbf{w}}) &= \mathrm{iHWT}(\mathbf{w}) - \mathrm{iHWT}(\widetilde{\mathbf{w}}) \\ &= \mathbf{X} - \widetilde{\mathbf{X}}. \end{aligned} \tag{19}$$

Combining the two equations yields

$$\|\mathbf{X} - \widetilde{\mathbf{X}}\|_F = \|\mathbf{w} - \widetilde{\mathbf{w}}\|_2, \tag{20}$$

which proves Eq. (16). □

Discussion. This result shows that the reconstruction error after top-k truncation is exactly controlled by the discarded coefficients. It also makes the trade-off between sparsity and fidelity transparent, because the distortion can be computed directly from coefficient magnitudes.

Proposition 3 (Deterministic Per-Subband Retention Guarantee of HWSP). *Let $\mathbf{W}$ be a weight*

matrix and let $\mathbf{w} = \mathrm{HWT}(\mathbf{W})$ *denote its two-dimensional Haar coefficients, partitioned into subbands* $S = \{\mathrm{LL}, \mathrm{LH}, \mathrm{HL}, \mathrm{HH}\}$. *Under* HWSP *with retention ratio* $\kappa \in (0, 1]$, *the number of retained coefficients in each subband* $s \in S$ *is*

$$k_s^{\mathrm{HWSP}} = \lfloor \kappa L_s \rfloor, \tag{21}$$

where L_s *is the number of coefficients in subband* s. *Therefore, for any designated subband* $s_0 \in S$ *and any target quota* $k_{s_0}^*$ *satisfying*

$$k_{s_0}^* \leq \lfloor \kappa L_{s_0} \rfloor, \tag{22}$$

HWSP *satisfies the quota deterministically:*

$$k_{s_0}^{\mathrm{HWSP}} \geq k_{s_0}^*. \tag{23}$$

Let the total retention budget of HWSP *be*

$$K^{\mathrm{HWSP}} = \sum_{s \in S} \lfloor \kappa L_s \rfloor. \tag{24}$$

By contrast, global magnitude pruning under the same total retention budget K^{HWSP} *does not, in general, provide a deterministic per-subband quota guarantee.*

Proof. By definition, HWSP applies the same retention ratio κ independently to each Haar subband. Hence, for every $s \in S$, the number of retained coefficients is exactly

$$k_s^{\mathrm{HWSP}} = \lfloor \kappa L_s \rfloor. \tag{25}$$

For any designated subband s_0, if the target quota satisfies

$$k_{s_0}^* \leq \lfloor \kappa L_{s_0} \rfloor, \tag{26}$$

then it follows directly that

$$k_{s_0}^{\mathrm{HWSP}} = \lfloor \kappa L_{s_0} \rfloor \geq k_{s_0}^*. \tag{27}$$

Therefore, HWSP deterministically satisfies the quota. The total number of retained coefficients under HWSP is

$$K^{\mathrm{HWSP}} = \sum_{s \in S} \lfloor \kappa L_s \rfloor. \tag{28}$$

Under global magnitude pruning, this same budget is allocated by a joint cross-subband ranking of coefficient magnitudes. Hence, there exist coefficient configurations for which the retained coefficients are concentrated in only a subset of subbands. Consequently, for a designated subband s_0, the retained count under global magnitude pruning can be smaller than a prescribed quota $k_{s_0}^* \leq \lfloor \kappa L_{s_0} \rfloor$. Therefore, global magnitude pruning does not, in general, provide a deterministic per-subband quota guarantee. □

C More Experimental Results

C.1 Language Modeling Results on OPT

We further evaluate HWSP on the OPT model series (1.3B, 6.7B, 13B) in Table 7. Across all three models, HWSP consistently outperforms magnitude pruning under both 20% and 40% sparsity, while remaining reasonably competitive with prior post-training pruning baselines.

C.2 Language Modeling Results on Qwen

To verify the generalization of our approach across different architectures, we further evaluate HWSP on the Qwen model family in Table 8 and Table 9. We observe that HWSP consistently outperforms magnitude pruning under both 20% and 40% sparsity levels, while maintaining competitive performance relative to SparseGPT and Wanda. Compared with magnitude pruning, our approach better preserves the reasoning and generative capabilities of Qwen at higher sparsity levels. Furthermore, HWSP achieves these results without requiring calibration data or weight updates, demonstrating its effectiveness and efficiency in compressing diverse model structures.

C.3 Uniform Subband Sparsity vs. LL Subband Preservation

To further investigate the role of different frequency components and support the design rationale of balanced subband-wise pruning, we compare two subband pruning strategies on LLaMA3-8B across multiple downstream tasks while controlling for overall sparsity. As detailed in Table 10, we evaluate perplexity on WikiText2 and C4, and accuracy on seven classification benchmarks. In the standard HWSP setting, a uniform sparsity ratio is applied to all four Haar subbands (LL, LH, HL, HH). In contrast, the HWSP (LL) variant fully preserves the LL subband (0% sparsity) while redistributing its budget to the three high-frequency subbands, resulting in more aggressive pruning of those components. As shown in Table 10, the standard HWSP strategy consistently outperforms the LL-preserved variant, with the performance gap widening under higher sparsity. These results indicate that over-preserving low-frequency components at the expense of high-frequency subbands degrades performance, particularly in aggressive pruning regimes. This finding supports the view that balanced preservation across frequency components is an effective

Sparsity	*Method*	Qwen2.5-7B			Qwen3-8B		
		WikiText2	**C4**	**PTB**	**WikiText2**	**C4**	**PTB**
0%	Dense	6.84	11.85	11.35	9.71	15.52	15.43
20%	SparseGPT	**6.89**	**11.92**	**11.44**	9.77	15.64	15.61
	Wanda	6.91	**11.92**	**11.44**	**9.71**	**15.58**	15.55
	Magnitude	7.80	13.33	12.87	10.16	15.98	15.83
	HWSP	6.99	12.04	11.56	9.85	15.70	15.80
40%	SparseGPT	**7.26**	**12.74**	**12.28**	10.30	16.98	16.85
	Wanda	7.42	12.77	12.35	**10.18**	**16.68**	**16.30**
	Magnitude	99.25	59.96	191.3	14.52	21.21	22.28
	HWSP	8.28	13.94	13.78	11.76	18.43	19.35

Table 8: Perplexity on WikiText2, C4, and PTB datasets for pruned Qwen2.5-7B and Qwen3-8B models under 20% and 40% sparsity. SparseLLM failed due to numerical instability during the Cholesky decomposition step.

		Method	**Openb.**	**ARC_e**	**ARC_c**	**WinoG.**	**HellaS.**	**PIQA**	**MathQA**	**Average↑**
Qwen2.5-7B	0%	Dense	33.60	80.72	47.70	73.40	60.00	78.73	43.32	59.64
	20%	SparseGPT	33.20	80.60	48.38	73.40	59.67	78.73	43.62	59.66
		Wanda	33.00	79.88	47.35	73.01	59.56	78.51	42.65	59.14
		Magnitude	33.20	78.70	46.84	70.09	57.12	77.80	41.37	57.87
		HWSP	31.80	80.30	47.87	72.30	59.60	78.67	41.34	58.84
	40%	SparseGPT	31.60	77.44	44.20	72.30	57.35	78.24	41.14	57.47
		Wanda	31.80	77.10	44.88	71.27	56.65	78.51	40.97	57.31
		Magnitude	28.20	51.94	26.11	55.56	32.72	57.94	26.10	39.80
		HWSP	31.80	77.61	44.71	68.51	56.74	78.35	40.60	56.90
Qwen3-8B	0%	Dense	31.20	83.53	55.46	68.11	57.13	76.66	49.31	60.20
	20%	SparseGPT	32.00	83.42	54.95	67.96	56.97	76.01	49.82	60.16
		Wanda	31.40	83.92	55.89	68.35	56.74	76.55	49.61	60.35
		Magnitude	30.00	81.52	52.73	68.19	56.97	76.12	46.77	58.90
		HWSP	31.40	83.12	54.44	68.90	56.57	75.90	47.44	59.68
	40%	SparseGPT	30.60	81.65	54.52	68.90	54.28	76.06	45.49	58.79
		Wanda	30.20	82.07	53.16	68.59	53.31	76.17	46.57	58.58
		Magnitude	27.80	76.56	46.67	65.59	51.90	73.99	38.12	54.38
		HWSP	29.00	75.76	46.67	65.27	52.52	75.52	41.04	55.11

Table 9: Accuracy (↑) of pruning methods on Qwen2.5-7B and Qwen3-8B by lm_eval v0.4.7.

design principle for maintaining the capabilities of large language models under sparsity constraints.

D Layer–wise Sensitivity Analysis

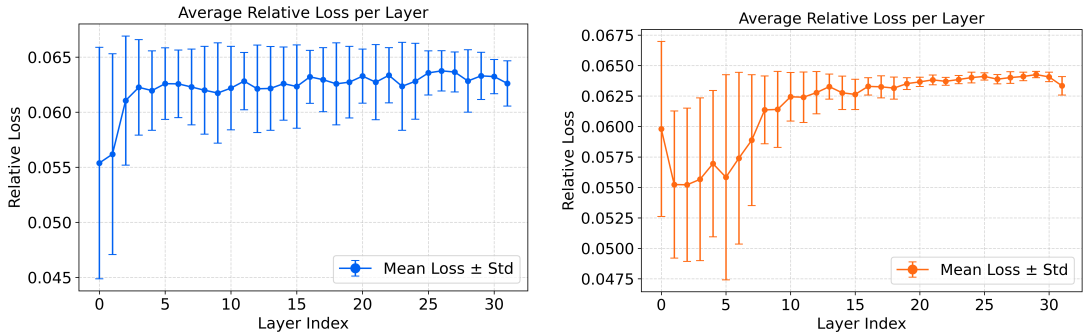
Since HWSP does not rely on calibration data, it enables efficient analysis of layer-wise pruning sensitivity. Specifically, for each linear projection W_ℓ in a Transformer layer, we compute the relative reconstruction error:

$$\Delta_\ell = \frac{\|W_\ell - \widetilde{W}_\ell\|_F}{\|W_\ell\|_F}, \tag{29}$$

where $\widetilde{W}_\ell$ denotes the Haar-pruned approximation of W_ℓ, and ℓ denotes the layer index. This metric measures layer sensitivity to pruning under a fixed sparsity ratio. Figure 2 presents the results for LLaMA–7B and OPT–6.7B at 20% sparsity. In both architectures, the first several layers exhibit higher reconstruction errors, indicating greater sensitivity to pruning in early layers. Sensitivity also differs across projections within each layer. MLP projections, such as `fc1`, `fc2`, and `mlp.up_proj`, are generally more sensitive than attention components, especially `k_proj` and `v_proj`. Overall, these sensitivity patterns provide empirical evidence for heterogeneous pruning robustness across layers and modules, and may be useful for designing more adaptive sparsity allocation strategies in

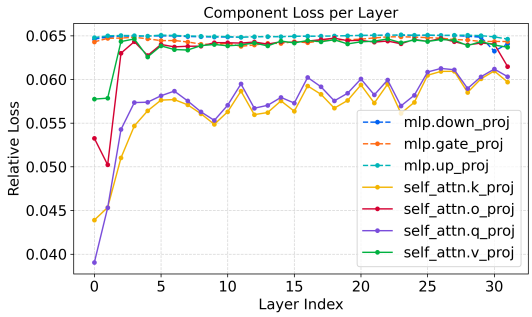
	Method	WikiText2	C4	Openb.	ARC_e	ARC_c	WinoG.	HellaS.	PIQA	MathQA	Average↑
0%	Dense	6.23	9.57	33.40	81.40	51.37	73.72	60.02	80.03	39.73	59.95
20%	HWSP	6.46	9.97	33.80	81.65	51.19	74.43	59.76	79.22	38.96	59.86
	HWSP (LL)	6.70	10.40	33.60	80.89	51.71	73.88	59.83	79.11	38.32	59.62
40%	HWSP	8.65	13.38	31.00	74.33	42.92	70.56	55.69	78.29	35.54	55.48
	HWSP (LL)	17.99	27.12	22.20	65.03	32.08	65.04	44.69	71.11	27.77	46.84

Table 10: Evaluation on LLaMA3-8B using lm_eval v0.4.7, reporting perplexity (↓) on WikiText2 and C4, and accuracy (↑) on different tasks.

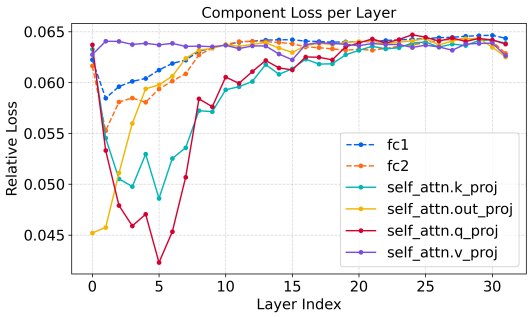


(a) *Llama-7B*

(b) *OPT-6.7B*



(c) *Llama-7B*



(d) *OPT-6.7B*

Figure 2: Layer-wise relative reconstruction loss (Δ_ℓ) after pruning 20% of weights with HWSP. Error bars (shaded regions) denote ± 1 *standard deviation* across all linear projections in the corresponding layer, revealing that the early layers are substantially more sensitive to sparsity.

future work.

E Qualitative Analysis on Multimodal Reasoning and Generation

To assess the effectiveness of HWSP in multimodal settings, we conduct qualitative comparisons with magnitude pruning on Janus-Pro-7B. As shown in Figure 3, under 40% sparsity, HWSP produces more accurate responses on multimodal reasoning tasks involving arithmetic computation, table-based retrieval, and scientific comparison. In contrast, magnitude pruning often produces incorrect or incomplete answers, suggesting that it is more likely to lose informative details under aggressive sparsity. We further compare text-to-image generation results under 20% and 40% sparsity. As shown in Figure 4, HWSP better preserves visual fidelity and semantic alignment with the input prompts, especially at higher sparsity levels. By comparison, magnitude pruning introduces more visible artifacts, including texture distortion, degraded text clarity, and color inconsistency. These examples suggest that HWSP is more effective in preserving both linguistic and visual information in multimodal tasks.

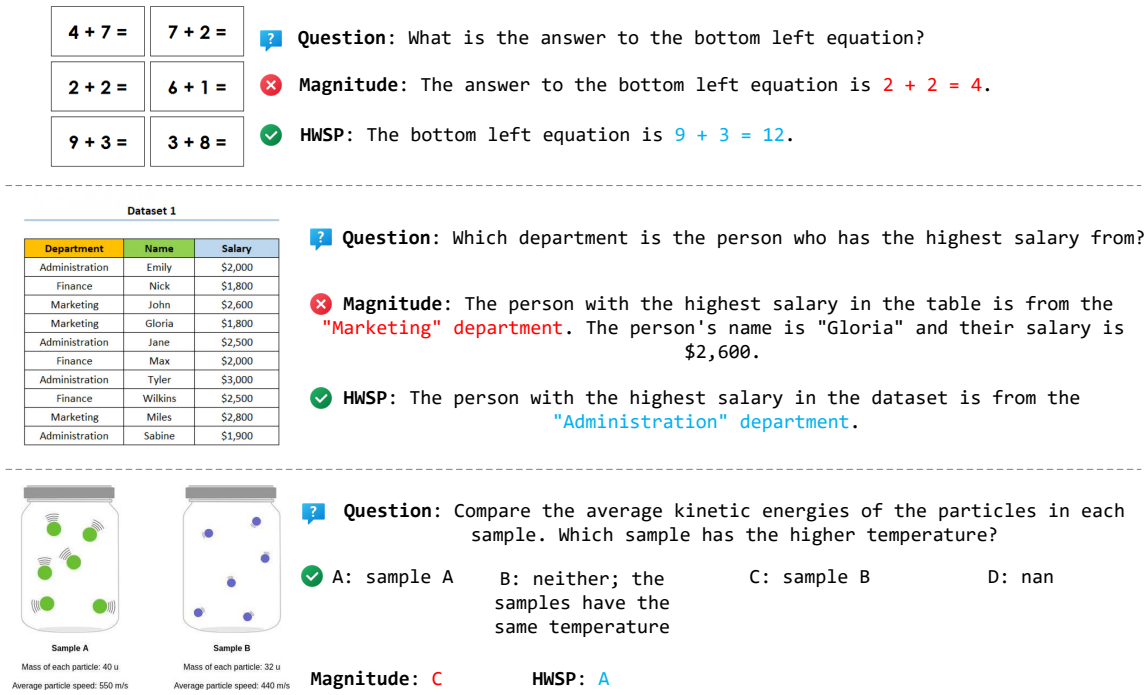


Figure 3: Case studies on multimodal reasoning tasks under 40% sparsity using Janus-Pro-7B.

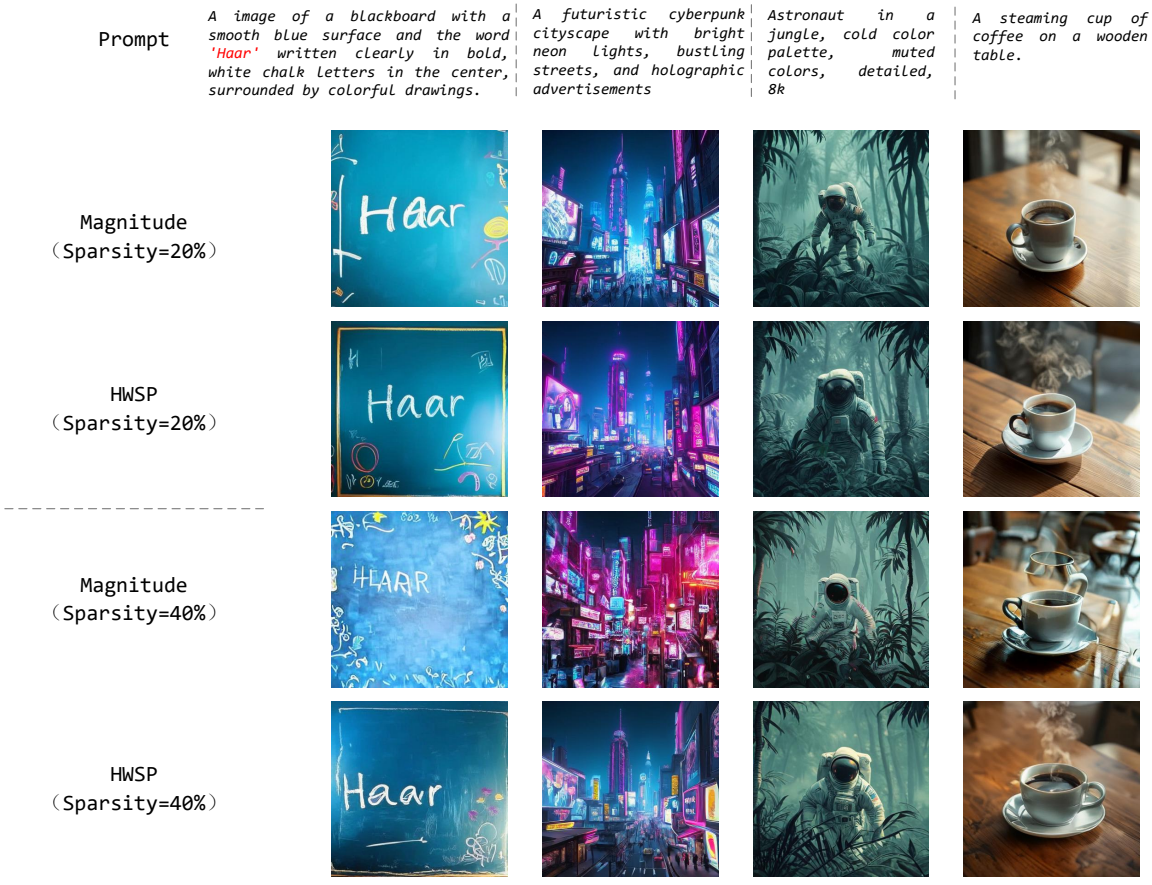


Figure 4: Text-to-image generation results under 20% and 40% sparsity using Janus-Pro-7B.