Dual Activation-Weight Sparsity: A Training-Free Framework for Efficient Large Language Model Compression

Luoyang Sun[1,2]*, Guanyan Li[1,2]*, Cheng Deng[3], Haifeng Zhang[1,2]†
Jian Zhao[4,5], Yongqiang Tang[1,2]†, Wensheng Zhang[6], Jun Wang[7]

[1]Institute of Automation, Chinese Academy of Sciences,
[2]University of Chinese Academy of Sciences,
[3]University of Edinburgh, [4]Zhongguancun Academy,
[5]Zhongguancun Institute of Artificial Intelligence, [6]Guangzhou University,
[7]University College London

Correspondence: sunluoyang2022@ia.ac.cn, liguanyan2022@ia.ac.cn

Abstract

Large language models (LLMs) excel at natural language tasks but face deployment challenges due to computational demands. We introduce **Dual Activation-Weight Sparsity (DAWS)**, a training-free framework that jointly exploits activation and weight sparsity through magnitude-based routing. Systematic analysis of pretrained transformers reveals two key observations (Figure 1): (1) the activation energy is concentrated in a few neurons, and (2) activation-weight contribution patterns exhibit substantial module-wise heterogeneity across attention and FFN modules. DAWS employs a three-tier routing strategy: high-magnitude activations pass through full-precision weights to preserve critical pathways, medium-magnitude activations use magnitude-pruned sparse weights for efficiency, and low-magnitude activations are directly discarded. Unlike prior work that uses activation-aware pruning methods like WANDA, our approach uses direct magnitude-based pruning, which we show is more robust to sample-level variations. Experiments on Llama and Mistral models demonstrate that DAWS maintains >98% of dense model performance at 50% sparsity, outperforming WANDA, TEAL, and R-Sparse.

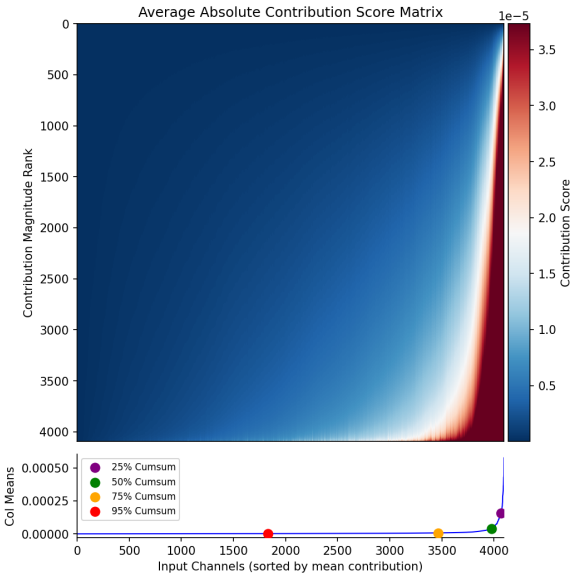

Figure 1: **Contribution analysis of activations and weights in linear layers.** Top: Heatmap showing the joint contribution of activation-weight pairs, revealing that high-magnitude activations paired with large-magnitude weights dominate the output. Bottom: Cumulative contribution of neurons sorted by activation magnitude. Approximately 15% of neurons contribute 75% of output energy. This motivates our three-tier routing strategy.

1 Introduction

Large language models (LLMs) have achieved remarkable success across natural language processing tasks, yet their massive parameter counts create severe deployment bottlenecks. Two complementary compression paradigms have emerged: *weight pruning*, which removes less important parameters offline, and *activation sparsity*, which dynamically skips computations on near-zero activations during the inference process.

Weight pruning has achieved impressive compression ratios. Magnitude-based methods like SparseGPT demonstrate that LLMs tolerate high sparsity with minimal accuracy degradation and no retraining (Frantar and Alistarh, 2023). WANDA and its extensions improve upon this by incorporating activation statistics into importance scoring (Sun et al., 2023; Yang et al., 2025). However, these methods produce static masks that cannot

* Equal Contribution.
† Corresponding author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 8350–8366
July 2-7, 2026

adapt to the input-dependent computational patterns inherent to transformer inference.

Activation sparsity methods take the opposite approach, exploiting the observation that many transformer activations contribute negligibly to outputs. TEAL achieves 40–50% model-wide sparsity through careful thresholding (Liu et al., 2024), while other work explores top-k selection or learned gating mechanisms. Yet these methods apply uniform computational policies regardless of which parameters process the activations, failing to account for varying parameter importance.

Recent work attempts to bridge these paradigms. R-Sparse partitions activations by magnitude and applies low-rank decompositions to approximate dense computations on residual activations (Zhang et al., 2025). However, this approach introduces SVD overhead and requires additional learned components, limiting its practicality. **This raises a critical design question: Can we jointly exploit activation heterogeneity and weight importance without costly decompositions or retraining?**

We answer affirmatively with **Dual Activation-Weight Sparsity (DAWS)**, motivated by two observations from systematic analysis of pretrained transformers:

O1: Extreme contribution asymmetry exists within activations. While heavy-tailed activation distributions are well-documented, we find the asymmetry is more pronounced than previously recognized. Approximately 15% of high-magnitude neurons contribute over 75% of output signal energy, while the remaining 80% contribute minimally (Figure 1). This suggests fundamentally different computational treatments are warranted for activations of different magnitudes.

O2: Module-wise heterogeneity of sparsity patterns. A comprehensive analysis across all transformer layers and projection types reveals that different weight matrices exhibit markedly different activation-weight contribution patterns; the degree of exploitable sparsity varies substantially across modules. This heterogeneity rules out a uniform sparsity policy and motivates module-specific budget allocation.

DAWS implements a three-tier routing architecture: high-magnitude activations pass through full-precision weights to preserve critical pathways, medium-magnitude activations use magnitude-pruned sparse weights for efficiency, and low-magnitude activations are directly discarded. Unlike activation-aware methods like WANDA, we use magnitude-based weight pruning which produces highly stable masks (IoU $>$0.92) suitable for pre-computation. This requires no retraining, no matrix decompositions, and no learned routers—only lightweight thresholding and pre-computed weight masks.

Contributions. We make three principal contributions:

1 Framework and theory. We propose DAWS, a training-free dual-sparsity framework with automated hierarchical search, and prove that under heavy-tailed activation distributions, DAWS achieves strictly lower approximation error than single-modality methods.

2 Empirical validation. Across Llama-3-8B, Llama-2-7B, and Mistral-7B, DAWS maintains $>$98% of dense performance at 50% sparsity, outperforming WANDA, TEAL, and R-Sparse by 1–3 percentage points.

3 Deployment insights. Through cross-sample stability analysis, we demonstrate that weight masks exhibit near-perfect consistency (IoU of 0.80-0.90,), validating offline pre-computation, while activation masks require dynamic computation (IoU of 0.20-0.35), informing practical implementation strategies.

2 Related Work

Weight Pruning. Weight pruning has become a central technique for LLM compression. Early work on magnitude-based pruning (Han et al., 2016) and the lottery ticket hypothesis (Frankle and Carbin, 2019) established that neural networks tolerate substantial parameter removal. For LLMs specifically, SparseGPT (Frantar and Alistarh, 2023) formulates pruning as layer-wise reconstruction, achieving high sparsity ratios with minimal accuracy loss. WANDA (Sun et al., 2023) and WANDA++ (Yang et al., 2025) improve upon this by incorporating activation statistics into importance scoring, demonstrating that activation-aware pruning outperforms pure magnitude-based methods. Structured pruning approaches like LLM-Pruner (Ma et al., 2023) remove entire attention heads or channels for better hardware efficiency. Despite strong compression rates, these methods produce static masks that cannot adapt to input-dependent computation patterns, limiting their runtime efficiency gains.

Activation Sparsity. Activation sparsity methods exploit the observation that many neurons in transformer models contribute negligibly to final outputs. ReLU-based approaches like Relufication (Mirzadeh et al., 2023) replace activation functions to induce natural sparsity, though this often requires architectural modifications. Threshold-based methods such as PowerInfer (Song et al., 2024) and TEAL (Liu et al., 2024) dynamically zero out small-magnitude activations, achieving 40–50% sparsity with carefully tuned thresholds. Top-k selection strategies like CATS (Lee et al., 2024) retain only the most salient activations per layer. GRIFFIN (Dong et al., 2024) focuses specifically on MLP sparsity, observing that feed-forward layers are particularly amenable to activation pruning. While these methods reduce FLOPs substantially, they apply uniform policies across all parameters regardless of weight importance.

Hybrid Sparsity Methods. Recent work explores joint optimization of weight and activation sparsity. R-Sparse (Zhang et al., 2025) partitions activations by magnitude and applies low-rank decompositions (SVD) to approximate computations on low-magnitude components, but introduces decomposition overhead and additional learned parameters. Amber (An et al., 2025) presents a training-free N:M activation sparsity method targeting the prefill stage, integrating structured activation pruning with post-training quantization for GPU acceleration. These compositional approaches demonstrate that combining multiple sparsity types can improve compression-accuracy tradeoffs, but existing methods either require retraining, introduce computational overhead through decompositions, or need learned routing mechanisms.

3 Method

3.1 Overview

DAWS consists of three components: (1) *dual-path routing* that partitions activations by magnitude and routes them through different weight matrices, (2) *hierarchical configuration search* that automatically determines optimal sparsity allocations, and (3) *offline weight pruning* using WANDA-style importance scoring.

Motivation from Observations. We analyze sparsity patterns in pretrained Llama-3-8B (Figure 6) and identify two quantitative observations that directly motivate our design:

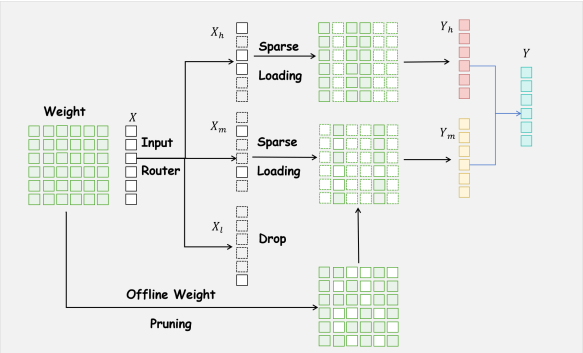

Figure 2: **DAWS architecture.** High-magnitude activations ($\mathbf{x}_h$) are routed through full-precision weights $\mathbf{W}$ to preserve critical pathways. Medium-magnitude activations ($\mathbf{x}_m$) pass through magnitude-pruned sparse weights $\mathbf{W}_p$ for efficiency. Low-magnitude activations ($\mathbf{x}_l$) are directly discarded. Offline weight pruning generates static masks $\mathbf{M}$ that remain fixed across inputs, while activation routing adapts dynamically based on per-sample magnitude thresholds τ_h and τ_l.

(O1) Extreme contribution asymmetry. Activations are heavy-tailed: only 15% of elements contribute more than 75% of the output energy, a sharper asymmetry than previously reported in the activation sparsity literature. This motivates a dedicated full-precision path for high-magnitude activations rather than uniform treatment.

(O2) Module-wise heterogeneity of sparsity patterns. A comprehensive analysis across all transformer layers and projection types (Figure 6, Appendix B.4) reveals that different weight matrices exhibit markedly different activation-weight contribution patterns: no two projection types share the same joint magnitude distribution, and the degree of exploitable sparsity varies substantially across modules. This rules out a single uniform sparsity ratio and motivates module-specific budget allocation.

These observations directly inform our three-tier routing and hierarchical search: (O1) justifies partitioning activations into high/medium/low tiers with different weight precision; (O2) justifies the per-module budget allocation in Stage 1. DAWS is thus the first framework that synergistically combines dynamic activation sparsity with static weight sparsity through principled magnitude-based routing, rather than SVD decomposition (R-Sparse) or learned routing.

3.2 Dual-Path Architecture

For a linear layer $\mathbf{y} = \mathbf{W}\mathbf{x}$ with $\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, we partition activations using two thresholds τ_h

and τ_l ($\tau_h > \tau_l$):

$$\begin{aligned} \mathbf{x}_h &= \mathbf{x} \odot \mathbf{1}_{\{|x_i| \geq \tau_h\}}, \\ \mathbf{x}_m &= \mathbf{x} \odot \mathbf{1}_{\{\tau_l \leq |x_i| < \tau_h\}}, \\ \mathbf{x}_l &= \mathbf{x} \odot \mathbf{1}_{\{|x_i| < \tau_l\}}, \end{aligned} \tag{1}$$

and compute the output as:

$$\mathbf{y} = \mathbf{W}\mathbf{x}_h + \mathbf{W}_p\mathbf{x}_m, \tag{2}$$

where $\mathbf{x}_l$ is directly discarded (treated as zeros), and $\mathbf{W}_p$ is a magnitude-pruned weight matrix obtained via:

$$\begin{aligned} s_{ij} &= |w_{ij}|, \\ M_{ij} &= \mathbf{1}_{\{s_{ij} \text{ in top-}k\}}, \\ \mathbf{W}_p &= \mathbf{W} \odot \mathbf{M}, \end{aligned} \tag{3}$$

where w_{ij} denotes the (i, j)-th entry of $\mathbf{W}$, and $M_{ij} \in \{0, 1\}$ is the corresponding binary mask retaining the top-k entries of $\mathbf{W}$ by magnitude. The thresholds τ_h and τ_l are set to the $(1 - \rho_h)$-th and $(1 - \rho_h - \rho_m)$-th percentiles of $|\mathbf{x}|$ to achieve high-activation ratio ρ_h and medium-activation ratio ρ_m, respectively.

Rationale for magnitude-based pruning. Unlike activation-aware methods like WANDA that incorporate activation statistics into pruning scores, we use direct magnitude-based pruning. Our analysis (Section 4.6) reveals that WANDA's activation-dependent pruning introduces sample-level bias, as the pruning masks vary significantly across different calibration samples (with minimum IoUs in the 0.65–0.75 range). In contrast, pure magnitude-based pruning produces consistent masks across samples, making it more suitable for our dynamic routing framework where activations already adapt to input variations. This complementarity—static weight masks with dynamic activation routing—provides both efficiency and adaptability.

This three-tier design ensures that critical activations receive full-precision computation, moderately important activations use efficient sparse computation, and negligible activations are eliminated entirely, balancing accuracy and efficiency.

3.3 Hierarchical Configuration Search

The key challenge is determining optimal (ρ_h, ρ_m, α) for each layer, where α denotes the weight density controlling the sparsity of $\mathbf{W}_p$. We employ a two-stage hierarchical search (Algorithm 1) that (1) allocates sparsity budgets across modules, then (2) optimizes intra-module activation-weight trade-offs. Here, a "module" refers to a distinct weight matrix in the model, specifically `q_proj`, `k_proj`, `v_proj`, `o_proj` in the attention block, and `up_proj`, `gate_proj`, `down_proj` in the FFN.

Stage 1: Inter-Module Allocation. We greedily allocate sparsity budgets across modules to minimize performance degradation. All modules are initialized with the same sparsity s_{init}. In each round, we tentatively increase the sparsity of each module $m \in \{q, k, v, o, \text{gate}, \text{up}, \text{down}\}$ by Δs_m and measure the resulting loss change ΔL_m. The increment is defined as:

$$\Delta s_m = \Delta s_{\text{base}} \cdot \frac{|W_{\min}|}{|W_m|}, \tag{4}$$

where $|W_m|$ denotes the parameter count of module m, $|W_{\min}| = \min_m |W_m|$ is the parameter count of the smallest module, and Δs_{base} is a base step size (set to 0.28; see Appendix A). Hence the increment is inversely proportional to module size, so larger modules advance more conservatively. After probing all modules, we permanently apply the sparsity increase to the least sensitive module $m^* = \arg\min_m \Delta L_m$, and repeat until the target overall sparsity is reached. This produces a per-module sparsity budget $\{s_m\}$.

Stage 2: Intra-Module Search. For each module with allocated sparsity budget s_m, we search for the optimal balance between activation sparsity s_a and weight sparsity s_w, which relate to the layer-wise parameters as follows: $\alpha = 1 - s_w$ (e.g., $\alpha = 0.2$ corresponds to $s_w = 0.8$, meaning 80% of weights are pruned), and $s_a = 1 - \rho_h$ (e.g., $s_a = 0.8$ means $\rho_h = 0.2$, i.e., 20% of activations are treated as high). The proportion of medium activations is given by $\rho_m = 1 - \rho_h - s_{\text{tail}}$, where s_{tail} denotes the proportion of discarded low activations (i.e., the ratio of elements in $\mathbf{x}_l$).

We search over a predefined set of candidate weight sparsities $s_w \in \mathcal{S}_w$. For each candidate s_w, the corresponding activation sparsity s_a is uniquely determined by the global compute budget constraint. Specifically, given target sparsity s_m and tail sparsity s_{tail}, the effective sparsity satisfies:

$$s_m = (s_a - s_{\text{tail}}) \cdot s_w + s_{\text{tail}}, \tag{5}$$

which yields the coupling function:

$$s_a = \frac{s_m - s_{\text{tail}}}{s_w} + s_{\text{tail}}. \tag{6}$$

For each valid (s_w, s_a) pair, we generate the corresponding weight mask via magnitude-based pruning and evaluate the resulting loss. The configuration (s_w^*, s_a^*) that minimizes the loss is then applied to the module.

3.4 Theoretical Analysis

We provide theoretical justification for DAWS's three-tier routing design and magnitude-based weight pruning strategy. Our analysis establishes formal error bounds and derives practical design guidelines.

3.4.1 Problem Setup

Assumption 1 (Heavy-Tailed Activations)**.** *There exist fractions $\rho_h, \rho_m \in (0,1)$ and concentration parameters β_h, β_m such that:*

$$\|\mathbf{x}_h\|^2 \geq \beta_h \|\mathbf{x}\|^2, \quad \beta_h > 0.5, \tag{7}$$

$$\|\mathbf{x}_h\|^2 + \|\mathbf{x}_m\|^2 \geq \beta_m \|\mathbf{x}\|^2, \quad \beta_m > 0.8. \tag{8}$$

Empirical Validation. We verified Assumption 1 on LLaMA3-8B by measuring concentration parameters across all layers and WikiText-2 tokens. With $\rho_h = 0.15$ and $\rho_m = 0.20$, the minimum observed values are $\min \|\mathbf{x}_h\|^2/\|\mathbf{x}\|^2 = 0.72 > 0.5 = \beta_h$ and $\min(\|\mathbf{x}_h\|^2 + \|\mathbf{x}_m\|^2)/\|\mathbf{x}\|^2 = 0.89 > 0.8 = \beta_m$, confirming the assumption holds with substantial margin. Strict concavity of $\beta_h(\rho_h)$ follows naturally as a derived property from this heavy-tailed structure.

Assumption 2 (Bounded Pruning Error)**.** *The magnitude-pruned weight matrix $\mathbf{W}_p$ with density α satisfies $\|\mathbf{W} - \mathbf{W}_p\|_F \leq \epsilon(\alpha)\|\mathbf{W}\|_F$, where $\epsilon(\alpha)$ decreases with α.*

Empirical Validation. For $w_{ij} \sim \mathcal{N}(0, \sigma^2)$ (standard in LLM initialization and approximately preserved after training), magnitude-based pruning with density α satisfies $\epsilon(\alpha) < \sqrt{1-\alpha} = \sqrt{\text{sparsity}}$, which is bounded and monotonically decreasing with α. We verified this on 11 representative weight matrices (attention and FFN projections from layers 0, 15, 31) of LLaMA3-8B (Table 1). At target 50% sparsity, $\max \epsilon = 0.26 \ll 0.71$; the bound holds at all sparsity levels.

Sparsity	Avg ϵ	Max ϵ	Bound $\sqrt{1-\alpha}$	Satisfies
10%	0.0162	0.0222	0.3162	✓
20%	0.0463	0.0629	0.4472	✓
30%	0.0873	0.1171	0.5477	✓
40%	0.1395	0.1828	0.6325	✓
50%	0.2044	0.2598	0.7071	✓
60%	0.2932	0.3482	0.7746	✓
70%	0.4001	0.4542	0.8367	✓
80%	0.5229	0.5826	0.8944	✓

Table 1: Empirical verification of Assumption 2.

3.4.2 Main Theoretical Result

Theorem 3 (Error Bound and Optimality of Three--Tier Routing)**.** *Under Assumptions 1 and 2:*

***(1)** The DAWS approximation error is bounded by*

$$\|\mathbf{y}_{DAWS} - \mathbf{y}\|^2 \leq 2\left[\epsilon^2(\alpha)+1\right](1-\beta_h)\,\|\mathbf{W}\|_F^2 \|\mathbf{x}\|^2. \tag{9}$$

***(2)** Under any fixed total sparsity budget, the optimal three-tier configuration achieves no higher error than the optimal two-tier configuration, since two-tier routing is a special case of three-tier routing ($\rho_m = 0$).*

The bound (9) separates the two error sources: $\epsilon^2(\alpha)$ captures pruning error on medium activations, and $(1-\beta_h)$ captures truncation error on low activations. Proof in Appendix C.

3.4.3 Superiority of Magnitude-Based Pruning

Proposition 4 (Static vs. Dynamic Weight Masks)**.** *Let $\mathbf{W}_p^{(j)}$ denote the activation-aware weight mask (e.g., WANDA) computed from a single calibration sample indexed by j, and let $\bar{\mathbf{W}}_p := \mathbb{E}_j[\mathbf{W}_p^{(j)}]$ be the population- averaged mask. Assuming the calibration sample j and test sample i are drawn independently, the expected approximation error when deploying a calibration-sample mask on test inputs decomposes as*

$$\mathbb{E}_{j,i}\|(\mathbf{W}_p^{(j)} - \mathbf{W})\mathbf{x}_m^{(i)}\|^2 \tag{10}$$

$$= \mathbb{E}_i\|(\bar{\mathbf{W}}_p - \mathbf{W})\mathbf{x}_m^{(i)}\|^2 + \mathbb{E}_{j,i}\|(\mathbf{W}_p^{(j)} - \bar{\mathbf{W}}_p)\mathbf{x}_m^{(i)}\|^2. \tag{11}$$

The second term is non-negative and vanishes if and only if $\mathbf{W}_p^{(j)}$ is constant in j.

This motivates our choice of scoring the weight mask by weight magnitude alone: incorporating activation statistics (as in WANDA) makes the mask depend on the calibration sample and introduces a non-zero variance term, whereas a magnitude-only mask is independent of any sample and has zero variance by construction. Empirically, the magnitude-only variant outperforms the WANDA

variant within DAWS (Table 5), consistent with the variance term being non-trivial in practice. Proof in Appendix C.

4 Experiments

4.1 Experimental Settings

Models and Datasets. We evaluate on Llama-3-8B (Dubey et al., 2024), Llama-2-7B (Touvron et al., 2023), and Mistral-7B (Jiang et al., 2023) using Hugging Face Transformers. We measure perplexity (PPL) on WikiText-2 (Merity et al., 2016) (2048 tokens) and evaluate on eight downstream tasks: Winogrande (WG) (Sakaguchi et al., 2020), PIQA (Bisk et al., 2020), ScienceQA (SciQ) (Welbl et al., 2017), OpenBookQA (OBQA) (Mihaylov et al., 2018), HellaSwag (HS) (Zellers et al., 2019), BoolQ (Clark et al., 2019), Arc-E, and Arc-C (Clark et al., 2018). For downstream tasks, higher is better; for PPL, lower is better. We report average accuracy across the eight tasks. All evaluations use the lm-eval (Gao et al., 2024) framework on A6000 GPUs with batch size 8.

Baselines. We compare against: **Relufication** (Mirzadeh et al., 2023) (activation replacement), **CATS** (Lee et al., 2024) and **GRIFFIN** (Dong et al., 2024) (MLP-only sparsity), **R-Sparse** (Zhang et al., 2025) (hybrid with SVD), **TEAL** (Liu et al., 2024) (pure activation sparsity). Methods marked with * indicate our reproductions for fair comparison.

4.2 Main Results

Table 2 presents comprehensive results at 50% sparsity across three model families. At this high sparsity level, DAWS achieves remarkable performance closely approaching dense models. On Llama-3-8B, Llama-2-7B, and Mistral-7B, our method achieves average scores of **68.15**, **64.90**, and **68.16**, respectively, compared to dense baseline scores of **69.46**, **65.88**, and **69.34**. This translates to retaining over 98% of original performance on average, demonstrating robustness and effectiveness across diverse architectures.

Beyond preserving dense model capabilities, DAWS significantly outperforms existing sparsity techniques. On Llama-3-8B, our average score of **68.15** surpasses the strongest competitor TEAL* by **1.71 points**, with consistent advantages across other models. The superiority extends to intrinsic language modeling: our PPL of **6.55** on Llama-3-8B is distinctly lower than TEAL*'s **6.80**, and we maintain this advantage on Mistral-7B (**5.47** vs. **5.56**). These results demonstrate that DAWS comprehensively preserves the model's linguistic functions while establishing a new state of the art for efficient sparse models.

4.3 Performance Across Sparsity Levels

As illustrated in Figure 3, we analyze the performance of R-Sparse, DAWS, and TEAL on the Llama-3-8B model across various sparsity ratios. Up to a sparsity of 50%, all methods demonstrate competitive and comparable performance, maintaining low Perplexity (PPL) and high average accuracy. However, a significant performance gap emerges at sparsity levels above 50%, where the methods begin to diverge significantly.

Beyond this 50% threshold, both R-Sparse and TEAL exhibit a rapid performance decline, with their PPL scores rising sharply and average accuracy dropping to approximately 45% at 70% sparsity. In stark contrast, DAWS showcases superior robustness, sustaining a much higher average accuracy of approximately 60% at 70% sparsity and showing a more graceful increase in PPL. This indicates that while all methods are effective at moderate sparsity, DAWS is significantly more adept at preserving model performance under high-compression scenarios.

4.4 Ablation Study

To investigate the individual contributions of our proposed components, we conduct an ablation study at 50% sparsity on Llama-3-8B, with the results presented in Table 3. Our baseline model, which relies solely on pure activation sparsity, achieves an average accuracy of 64.85%. Upon incorporating the inter-module sparsity allocation (S1), the performance significantly improves, boosting the average accuracy by 1.79%. Adding the intra-module configuration search (S2) to the baseline provides an even more substantial gain, increasing the average accuracy by 2.60%.

Our full model, which integrates both S1 and S2 components, achieves the best performance across all evaluated benchmarks. It reaches the highest average accuracy of 67.83%, marking a total improvement of 2.98% over the baseline, and attains the lowest PPL of 7.17. These results clearly demonstrate that both the inter-module and intra-module strategies are effective and contribute positively. The combined effect of these components is synergistic, leading to the optimal performance and

Method	PPL	WG	PIQA	SciQ	OBQA	HS	BoolQ	Arc-E	Arc-C	Avg
Llama-3-8B										
Baseline	6.13	72.85	79.71	96.2	34.8	60.17	81.44	80.09	50.43	69.46
Relufication	$4.59E^5$	50.91	53.54	22.3	14.8	25.52	52.84	24.37	21.59	33.23
R-Sparse*$_{0\%}$	8.53	67.64	75.14	95.7	31.2	54.22	76.91	76.56	42.83	65.03
TEAL*$_{0\%}$	8.53	65.59	75.73	95.7	31.2	54.33	76.61	76.56	43.09	64.85
DAWS$_{0\%}$	**7.17**	**70.88**	**77.91**	**95.7**	**34.4**	**58.62**	**80.09**	**78.75**	**46.25**	**67.83**
CATS	–	48.22	56.96	36.9	16.2	27.3	49.05	30.3	22.35	35.91
GRIFFIN	–	52.8	64.74	73.9	19.2	35.62	47.61	43.64	23.38	45.11
R-Sparse$_{50\%}$	–	69.3	77.69	96.0	31.6	56.64	76.73	76.94	44.71	66.20
R-Sparse*$_{50\%}$	6.82	69.69	77.64	96.0	32.2	56.46	76.61	76.14	45.39	66.27
TEAL*$_{50\%}$	6.80	69.38	78.18	95.3	33.4	56.51	76.67	76.94	45.14	66.44
DAWS$_{50\%}$	**6.55**	**72.53**	**77.69**	**96.4**	**35.4**	**58.45**	**79.66**	**77.19**	**47.87**	**68.15**
Llama-2-7B										
Baseline	5.47	69.14	78.07	93.8	31.4	57.14	77.74	76.35	43.43	65.88
Relufication	$2.54E^4$	49.41	54.19	25.9	15.4	25.83	60.03	27.9	24.15	35.35
R-Sparse*$_{0\%}$	7.21	68.75	73.72	93.8	27.0	49.22	71.44	71.63	38.65	61.78
TEAL*$_{0\%}$	6.83	66.93	76.71	93.9	29.8	53.68	74.40	73.57	39.85	63.61
DAWS$_{0\%}$	**5.97**	**69.53**	**77.53**	**93.2**	**30.2**	**55.27**	**74.68**	**74.49**	**42.32**	**64.65**
CATS	–	55.01	66.97	57.2	20.2	36.27	62.81	44.02	27.56	46.26
GRIFFIN	–	53.59	64.74	77.7	17.4	35.64	56.42	40.74	21.08	45.91
R-Sparse$_{50\%}$	–	67.4	77.31	93.9	31.4	54.26	72.84	74.58	40.78	64.06
R-Sparse*$_{50\%}$	5.82	67.48	76.39	93.5	29.2	54.11	72.63	74.24	39.85	63.43
TEAL*$_{50\%}$	5.75	67.64	77.20	93.4	30.8	54.55	73.06	74.24	40.36	63.91
DAWS$_{50\%}$	**5.65**	**67.88**	**77.53**	**93.4**	**32.0**	**55.86**	**75.44**	**75.38**	**41.72**	**64.90**
Mistral-7B										
Baseline	5.32	73.88	80.20	95.9	33.2	60.89	82.14	79.63	48.89	69.34
Relufication	$3.31E^4$	47.91	51.47	23.4	15.2	25.79	41.56	27.31	22.44	31.89
R-Sparse*$_{0\%}$	6.54	69.38	79.00	95.9	28.8	57.18	79.69	76.98	44.03	66.37
TEAL*$_{0\%}$	6.85	69.38	79.43	96.3	28.8	58.58	81.28	77.53	45.99	67.16
DAWS$_{0\%}$	**5.74**	**69.85**	**79.22**	**96.1**	**31.4**	**59.30**	**80.61**	**79.50**	**46.93**	**67.86**
CATS	–	50.83	58.05	28.6	19.2	28.06	60.21	30.22	25.43	37.58
GRIFFIN	–	54.22	67.90	79.1	19.8	39.99	47.31	49.83	26.37	48.07
R-Sparse$_{50\%}$	–	72.69	79.92	96.1	30.6	58.94	82.81	78.91	47.18	68.39
R-Sparse*$_{50\%}$	5.71	69.61	79.33	96.1	31.0	57.91	79.60	77.69	45.82	67.13
TEAL*$_{50\%}$	5.56	70.40	79.00	96.3	30.4	58.69	81.65	78.54	46.50	67.69
DAWS$_{50\%}$	**5.47**	**70.88**	**79.76**	**95.8**	**32.4**	**59.79**	**81.10**	**78.87**	**46.67**	**68.16**

Table 2: Performance at 50% sparsity across three model families. DAWS consistently outperforms baselines while maintaining >98% of dense performance. The subscript ($X\%$) indicates the ratio of pre-filled tokens during evaluation. Methods marked with * are our reproductions for fair comparison and "—" means the metric was not reported in source paper.

validating our overall design.

4.5 Dynamic Activation Routing Analysis

Cross-Sample Activation Variability. To validate the necessity of dynamic activation thresholding, we measure the IoU of activation masks across random sample pairs from the WikiText-2 validation set. Figure 4 reveals that activation patterns are profoundly input-dependent. The cross-sample IoU for activation masks is extremely low, generally ranging from 0.20 to 0.35, which indicates very high variability.

This extreme variability across samples, unequivocally demonstrates that activation patterns are fundamentally dynamic and cannot be captured by a single, static mask. The consistently low IoU values underscore that any attempt to use a precomputed activation sparsity pattern would fail, confirming the critical need for a per-sample, adaptive thresholding mechanism.

Impact of Fixed vs Dynamic Masking. To quantify the performance cost of using fixed activation masks, we conduct an ablation where we precompute activation masks on calibration data and apply them statically during inference. Table 4 shows the results at 50% sparsity on Llama-3-8B. As the results demonstrate, substituting dynamic

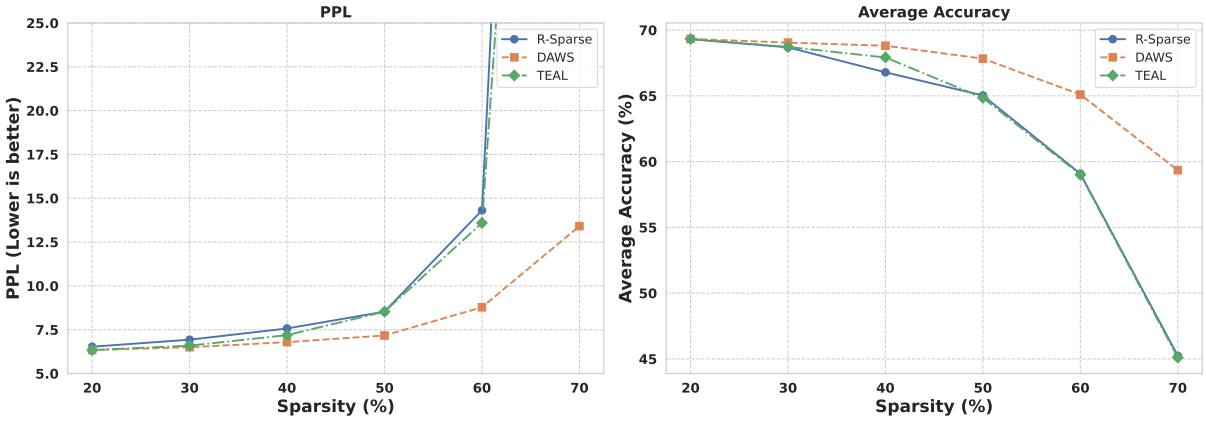

Figure 3: Comparative performance of different methods across multiple sparsity ratios.

Configuration	PPL	WG	PIQA	SciQ	OBQA	HS	BoolQ	Arc-E	Arc-C	Avg
Baseline (pure activation)	8.53	65.59	75.73	95.7	31.2	54.33	76.61	76.56	43.09	64.85
+ S1 (inter-module)	7.64	68.82	77.31	96.2	34.2	57.44	75.72	78.03	45.39	66.64
+ S2 (intra-module)	7.37	71.82	77.42	95.3	33.2	57.47	79.11	78.79	46.5	67.45
+ S1 + S2 (full)	**7.17**	**70.88**	**77.91**	**95.7**	**34.4**	**58.82**	**80.09**	**78.75**	**46.25**	**67.83**

Table 3: Ablation study at 50% sparsity on Llama-3-8B. **Baseline**: pure activation sparsity without weight pruning. **S1**: adds inter-module sparsity allocation. **S2**: adds intra-module configuration search. Both stages contribute significantly (S1: +1.79%, S2: +2.60%), with synergistic effects when combined (+2.98% total).

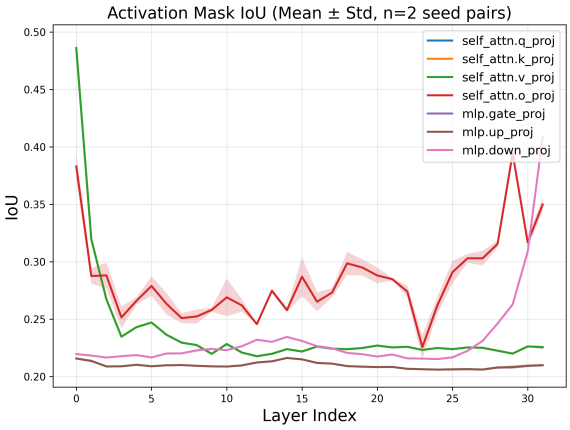

Figure 4: **Cross-sample stability of activation masks.** Activation mask IoU across random sample pairs shows substantial variability.

masks with their static counterparts causes a severe performance drop, with accuracy falling from 68.15 to 52.36. This confirms that a pre-computed activation pattern is insufficient, and confirms that dynamic, per-sample activation thresholding is essential for maintaining model quality.

4.6 Weight Pruning Strategy Analysis

Sample-Level Consistency of Pruning Methods. A critical question for hybrid sparsity methods is whether weight pruning should incorporate activation statistics (as in WANDA) or use pure magnitude-based pruning. We investigate this by measuring the Intersection-over-Union (IoU) of weight masks generated across random sample pairs from the WikiText-2 dataset.

Configuration	PPL	Avg Acc	Δ vs Dense
Dense Baseline	6.13	69.46	–
DAWS (Dynamic)	**6.55**	**68.15**	**-1.31**
DAWS (Static Mask)	55.28	52.36	-17.10

Table 4: Performance comparison of dynamic versus static activation mask at 50% sparsity on Llama-3-8B.

Figure 5 reveals the full extent of WANDA's instability. Even its best-performing components only achieve an IoU of 0.80-0.90, meaning 10-20% of the weight mask still differs between samples. This instability becomes far more severe in the attention output projection ($self\_attn.o\_proj$), which exhibits much larger sample-dependent variation. In stark contrast, magnitude-based pruning is data-independent and thus perfectly stable. This finding validates our design choice to use a reliably stable magnitude mask as the structural foundation for DAWS. It provides the necessary predictability that WANDA's component-specific instability lacks, complementing our method's dynamic activation sparsity.

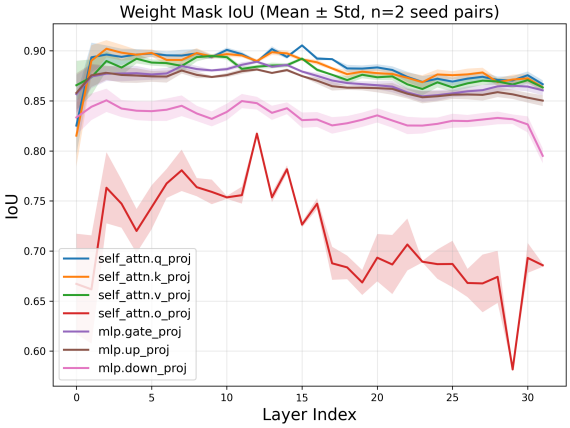

Figure 5: **Cross-sample stability of weight pruning methods.** Comparing the mask IoU of WANDA-style activation-aware pruning across different samples.

Performance Comparison. To determine the optimal weight pruning strategy, we compare DAWS with magnitude-based versus WANDA-style pruning. Table 5 shows that magnitude-based pruning is superior at 50% sparsity, outperforming WANDA on both PPL (7.17 vs. 7.30) and average accuracy (67.83% vs. 67.18%).

We attribute this to a core principle of hybrid sparsity design: since dynamic activation routing already adapts to input-specific patterns, the weight pruning component should provide a stable structural foundation. Magnitude-based masks fulfill this role with perfect consistency, while WANDA's data-dependent masks introduce a conflicting source of dynamism that creates redundancy and instability. This establishes that for hybrid sparsity methods with dynamic activation routing, stable magnitude-based weight pruning is preferable to activation-aware methods like WANDA.

Method	PPL	Avg Acc	Δ vs Dense
Dense Baseline	6.13	69.46	–
DAWS (Magnitude)	**7.17**	**67.83**	**-1.63**
DAWS (WANDA)	7.30	67.18	-2.28

Table 5: Performance comparison of DAWS using magnitude-based versus WANDA-style weight pruning at 50% sparsity on Llama-3-8B.

Complementarity of Static Weights and Dynamic Activations. These findings reveal an elegant complementarity at the heart of DAWS. Weight importance, derived from magnitude pruning, is input-invariant, allowing it to form a stable, pre-computed structural backbone. In contrast, activation importance is profoundly input-dependent, necessitating dynamic, runtime computation. This sharp division of labor is a core design feature, not a limitation. It allows DAWS to achieve input-adaptive efficiency through lightweight dynamic activation thresholding, while avoiding the overhead and component-specific instability of dynamic weight selection. This combination of a static foundation and dynamic execution provides both adaptability and simplicity, making DAWS a highly practical solution.

5 Conclusion

We introduce DAWS, a training-free framework that synergistically combines activation and weight sparsity via magnitude-based routing. Motivated by complementary sparsity patterns across layers, DAWS routes high-magnitude activations through full-precision weights and low-magnitude ones through sparse weights, with hierarchical search optimizing layer-wise configurations. Theoretical analysis proves DAWS outperforms single-modality sparsity under heavy-tailed distributions. Experiments on Llama and Mistral models show >98% dense performance at 50% sparsity. Stability analysis confirms pre-computable weight masks (IoU >0.92) and dynamic activation masks enable practical deployment. DAWS offers a simple, effective solution for efficient LLM inference.

Limitations and Future Work. While DAWS demonstrates strong results, several directions warrant investigation: (1) extending to joint sparsity-quantization for additional compression, (2) hardware-aware search optimizing for specific accelerators, (3) adaptive runtime thresholding based on input characteristics, (4) application to emerging architectures like mixture-of-experts models.

6 Acknowledgment

This work was supported by the National Natural Science Foundation of China (62476274, U22B2048).

References

Tai An, Ruwu Cai, Yanzhe Zhang, Yang Liu, Hao Chen, Pengcheng Xie, Sheng Chang, Yiwu Yao, and Gongyi Wang. 2025. Amber pruner: Leveraging n:m activation sparsity for efficient prefill in large language models. *Preprint*, arXiv:2508.02128.

Yonatan Bisk, Rowan Zellers, Ronan Le bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proc. AAAI*, page 7432–7439.

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

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*.

Harry Dong, Beidi Chen, and Yuejie Chi. 2024. Prompt-prompted mixture of experts for efficient llm generation. *CoRR*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *Preprint*, arXiv:1803.03635.

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

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

Song Han, Huizi Mao, and William J. Dally. 2016. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *Preprint*, arXiv:1510.00149.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Donghyun Lee, Je-Yong Lee, Genghan Zhang, Mo Tiwari, and Azalia Mirhoseini. 2024. Cats: Contextually-aware thresholding for sparsity in large language models. *arXiv preprint arXiv:2404.08763*.

James Liu, Pragaash Ponnusamy, Tianle Cai, Han Guo, Yoon Kim, and Ben Athiwaratkun. 2024. Training-free activation sparsity in large language models. *arXiv preprint arXiv:2408.14690*.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *Preprint*, arXiv:2305.11627.

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. In *Proc. EMNLP*.

Iman Mirzadeh, Keivan Alizadeh, Sachin Mehta, Carlo C Del Mundo, Oncel Tuzel, Golnoosh Samei, Mohammad Rastegari, and Mehrdad Farajtabar. 2023. Relu strikes back: Exploiting activation sparsity in large language models. *arXiv preprint arXiv:2310.04564*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *Proc. AAAI*, page 8732–8740.

Yixin Song, Zeyu Mi, Haotong Xie, and Haibo Chen. 2024. Powerinfer: Fast large language model serving with a consumer-grade gpu. *Preprint*, arXiv:2312.12456.

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

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*.

Johannes Welbl, Nelson F Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106.

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. 2025. Wanda++: Pruning large language models via regional gradients. *arXiv preprint arXiv:2503.04992*.

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

Zhenyu Zhang, Zechun Liu, Yuandong Tian, Harshit Khaitan, Zhangyang Wang, and Steven Li. 2025. R-sparse: Rank-aware activation sparsity for efficient llm inference. In *The Thirteenth International Conference on Learning Representations*.

A Implementation Details

A.1 Searching Details.

Calibration uses 512 tokens from WikiText-2 dataset. For Stage 1, we use $s_{\text{init}} = 0.3$ and $\Delta s_{\text{base}} = 0.28$. For Stage 2, we search over $s_w \in [0.75, 1.0]$ with an increment of 0.025, yielding 11 candidates per module; a value of $s_w = 1.0$ corresponds to pure activation sparsity with no weight pruning. In total, the search requires approximately $7 \times 11 \times L$ forward passes, where L is the number of layers, completing in under 10 minutes per model on an A6000 GPU. The step-by-step procedure is detailed in Algorithm 1.

Algorithm 1 Hierarchical Search for Sparsity Configuration

Input: Model M, Target sparsity S_{target}, Calibration data D_{cal}
Output: Configured sparse model with optimal $\{(s_w^m, s_a^m)\}$
// Stage 1: Inter-Module Sparsity Allocation
Initialize $s_m \leftarrow s_{\text{init}}$ for all modules m
Compute baseline loss L_{base} on D_{cal}
while get_avg_sparsity($\{s_m\}, M$) $< S_{\text{target}}$ **do**
    for each module m **do**
        $\Delta s_m \leftarrow \Delta s_{\text{base}} \cdot |W_{\min}|/|W_m|$ ▷ Parameter-aware increment
        Temporarily set $s_m \leftarrow s_m + \Delta s_m$
        Evaluate loss L_m on D_{cal}
        Compute sensitivity $\Delta L_m \leftarrow L_m - L_{\text{base}}$
        Restore s_m to original value
    end for
    $m^* \leftarrow \arg\min_m \Delta L_m$ ▷ Select least sensitive
    $s_{m^*} \leftarrow s_{m^*} + \Delta s_{m^*}$ ▷ Permanently update
    Update L_{base} with new configuration
end while
Output sparsity budget dictionary $\{s_m\}_{\text{budgets}}$
// Stage 2: Intra-Module Configuration Search
for each module m in M **do**
    Retrieve allocated sparsity budget s_m from Stage 1
    Compute weight importance: $I(w_{ij}) = |w_{ij}|$
    Initialize $L_{\min} \leftarrow \infty$, $s_w^* \leftarrow 0$, $s_a^* \leftarrow 0$
    for each candidate weight sparsity $s_w \in \mathcal{S}_w$ **do** ▷ Grid search over weight sparsity
        Compute s_a^{cand} from candidate s_w using Eq. (6)
        Generate weight mask $\mathbf{M}$ using top-s_w of $I(W_m)$
        Set activation threshold for sparsity s_a^{cand}
        Evaluate loss L on D_{cal}
        if $L < L_{\min}$ **then**
            $L_{\min} \leftarrow L$
            $s_w^* \leftarrow s_w$
            $s_a^* \leftarrow s_a^{\text{cand}}$ ▷ Save the corresponding activation sparsity
        end if
    end for
    Permanently apply configuration (s_w^*, s_a^*) to module m
end for
return Configured model M

A.2 Threshold Calibration

In line with the R-Sparse implementation, we pre-compute percentile statistics from the activation magnitude distribution of each layer on the Wikitext-2 dataset. During inference, this allows us to

determine sparsity thresholds through simple linear interpolation of the stored statistics. This strategy entirely avoids expensive on-the-fly sorting, reducing the threshold computation overhead to a constant $O(1)$ per layer.

B More detailed experimental results

B.1 Sparsity Levels

This section provides the comprehensive, per-benchmark results that support the main analysis in this paper. Table 6 details the performance of R-Sparse, TEAL, and our method, DAWS, across all evaluated benchmarks for sparsity ratios ranging from 20% to 70%. This granular data provides the underlying foundation for the summarized figures and analysis presented in the main body. The trends observed in this table are consistent with our primary conclusion that DAWS maintains superior performance, particularly at higher sparsity levels, with particularly strong results at 40–60% sparsity. At 70% sparsity, all methods degrade significantly, but DAWS degrades most gracefully.

Sparsity	PPL	WG	PIQA	SciQ	OBQA	HS	BoolQ	Arc-E	Arc-C	Avg
R-Sparse										
20%	6.53	74.11	79.27	96.1	36.4	59.99	80.46	80.35	47.78	69.31
30%	6.93	73.16	78.45	96.2	35.8	58.83	79.69	79.84	47.53	68.69
40%	7.57	70.48	77.04	96.1	32.6	57.28	79.33	77.36	44.11	66.79
50%	8.53	67.64	75.14	95.7	31.2	54.22	76.91	76.56	42.83	65.03
60%	14.31	64.56	71.11	94.1	24.6	46.27	68.41	68.35	35.24	59.08
70%	126.25	52.80	59.09	81.6	15.8	28.90	60.98	43.56	19.11	45.23
TEAL										
20%	6.34	73.40	79.22	96.1	35.2	60.01	80.40	80.47	49.83	69.33
30%	6.59	72.77	78.62	95.6	34.00	59.68	80.86	79.88	48.38	68.72
40%	7.19	72.77	78.40	96.2	31.8	57.96	80.40	79.50	46.33	67.92
50%	8.53	65.59	75.73	95.7	31.2	54.33	76.61	76.56	43.09	64.85
60%	13.59	60.30	72.31	92.8	27.2	45.95	68.90	70.24	34.39	59.01
70%	92.72	50.67	61.37	78.3	16.4	29.03	54.74	48.78	21.50	45.10
DAWS										
20%	6.34	73.40	79.22	96.1	35.2	60.01	80.40	80.47	49.83	69.33
30%	6.50	73.40	78.84	96.0	34.4	59.71	81.01	80.01	49.06	69.05
40%	6.79	72.14	79.00	96.5	34.2	58.97	80.83	79.88	48.98	68.81
50%	7.17	70.88	77.91	95.7	34.4	58.62	80.09	78.75	46.25	67.83
60%	8.78	69.53	76.71	94.8	31.6	54.58	77.25	75.04	41.30	65.10
70%	13.41	64.01	72.03	92.4	26.6	47.44	70.40	66.96	34.98	59.35

Table 6: Performance at different sparsity levels on Llama-3-8B. DAWS maintains superior performance across all sparsity ratios, with particularly strong results at 40–60% sparsity. At 70% sparsity, all methods degrade significantly, but DAWS degrades most gracefully.

B.2 Scalability Across Model Sizes

To verify the scalability of DAWS across different model sizes, we evaluate performance on both LLaMA-2-13B and Qwen2.5-1.5B across sparsity levels ranging from 20% to 70%.

Results on LLaMA-2-13B. As shown in Table 7, DAWSperforms robustly on the 13B model, maintaining competitive performance up to 50% sparsity with only marginal degradation. At higher sparsity levels (70%), performance drops more noticeably, which is expected given the aggressive compression ratio.

Results on Qwen2.5-1.5B. As shown in Table 8, the smaller 1.5B model exhibits lower inherent redundancy, making it more sensitive to aggressive sparsity. DAWSmaintains competitive performance up to 50% sparsity, but degradation becomes more pronounced at 60%–70% sparsity compared to the larger model. This suggests that the benefits of activation-weight co-sparsity are more significant in larger, more redundant models.

Overall, these results confirm that DAWSgeneralizes effectively across different model scales, with graceful performance degradation as sparsity increases.

Sparsity	PPL	WG	PIQA	SciQ	OBQA	HS	BoolQ	Arc-E	Arc-C	Avg
Baseline	4.88	72.22	79.00	94.5	35.2	60.07	80.61	79.38	48.46	68.68
20%	4.98	72.22	79.11	94.8	36.0	59.88	80.28	79.50	48.04	68.73
30%	5.04	70.96	78.67	95.5	34.0	59.69	80.15	79.88	47.87	68.34
40%	5.14	71.35	78.02	95.4	34.0	59.38	79.97	79.21	47.78	68.14
50%	5.31	69.69	78.24	95.3	33.8	58.91	80.40	78.11	46.93	67.67
60%	5.78	69.46	77.26	95.6	31.6	57.42	78.75	76.81	44.11	66.38
70%	7.93	65.43	74.86	93.7	26.2	50.87	75.63	71.34	37.71	61.97

Table 7: Performance of LLaMA-2-13B across sparsity levels.

Sparsity	PPL	WG	PIQA	SciQ	OBQA	HS	BoolQ	Arc-E	Arc-C	Avg
Baseline	9.26	63.61	75.68	94.8	32.2	50.22	72.91	75.55	41.21	63.27
20%	9.37	64.25	75.73	94.5	31.6	50.03	73.15	75.51	40.96	63.22
30%	9.56	63.93	75.63	93.9	31.4	49.49	72.84	74.20	40.96	62.79
40%	9.90	62.83	74.37	93.8	30.2	48.68	73.18	73.61	40.96	62.20
50%	10.67	63.38	73.39	93.9	28.2	46.42	71.62	72.26	39.85	61.13
60%	13.01	61.33	71.06	93.7	27.2	43.79	68.69	69.82	36.18	58.97
70%	26.52	56.43	66.59	90.4	22.6	37.78	63.85	59.93	29.35	53.37

Table 8: Performance of Qwen2.5-1.5B across sparsity levels.

B.3 Robustness to Calibration Data

To assess the sensitivity of DAWSto the choice of calibration data, we evaluate performance under three different calibration sets: WikiText-2, C4, and MathQA, with results reported in Table 9.

Despite the distinct domain distributions of these calibration corpora, downstream task performance remains highly consistent across all settings, with average accuracy fluctuating by only 0.42%. This robustness indicates that the hierarchical configuration search effectively captures the intrinsic activation and weight redundancy of the model, rather than overfitting to the specific calibration distribution.

Calibration Set	WikiText2	WG	PIQA	SciQ	OBQA	HS	BoolQ	ARC-E	ARC-C	Average	C4 (PPL)	MathQA
Baseline	6.13	72.85	79.71	96.2	34.8	60.17	81.44	80.09	50.43	69.46	9.31	40.67
WikiText-2	7.21	71.59	78.29	95.7	33.2	58.2	78.23	77.86	48.89	67.75	11.49	36.52
C4	6.99	70.09	78.29	96.0	35.8	58.64	79.39	78.45	45.9	67.82	11.07	37.25
MathQA	7.18	71.43	78.94	95.9	34.0	57.89	77.46	77.1	46.5	67.40	11.44	35.78

Table 9: Impact of calibration set choice on DAWS performance on Llama-3-8B at 50% sparsity. Results show that performance is robust across different calibration datasets, with WikiText-2 performing competitively across all benchmarks.

B.4 Visualization

Figure 6 presents a comprehensive visualization of activation-weight contribution patterns across all 32 layers and different module types in Llama-3-8B. Each subplot shows the cumulative contribution matrix for a specific layer and projection type (q_proj, k_proj, v_proj, o_proj for attention, and gate_proj, up_proj, down_proj for FFN).

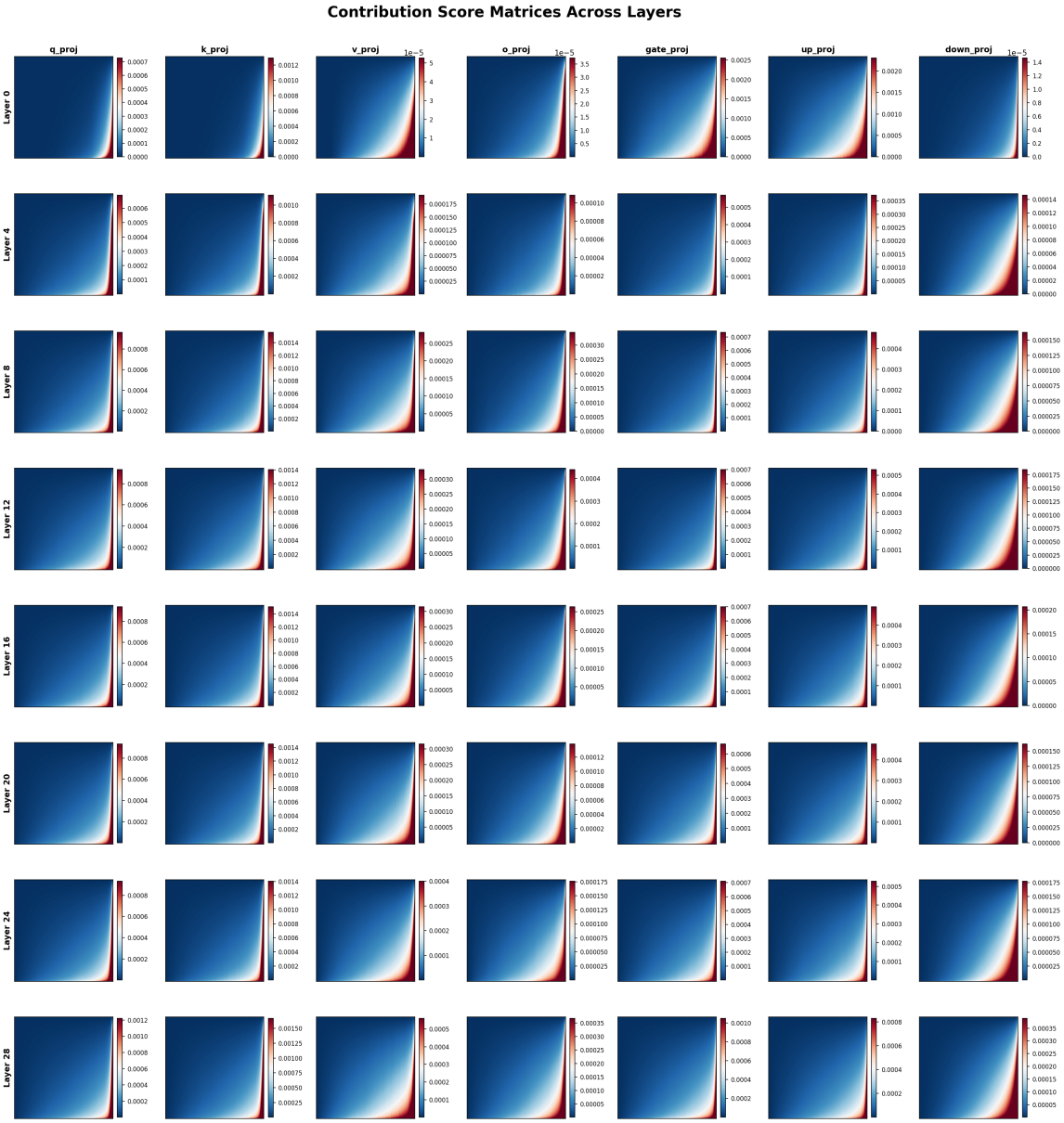

Figure 6: **Comprehensive layer-wise contribution analysis for Llama-3-8B.** Each subplot shows the cumulative contribution of activation-weight pairs for a specific layer and module type. Rows correspond to layers 0, 4, 8, 12, 16, 20, 24, 28 (sampled every 4 layers). Columns show different projection types: q_proj, k_proj, v_proj, o_proj (attention), gate_proj, up_proj, down_proj (FFN). The blue-to-red gradient indicates contribution magnitude, with red regions representing high contribution. Consistent heavy-tailed patterns across all modules validate our three-tier routing design. The steeper gradients in FFN projections (rightmost columns) confirm that FFN layers exhibit more extreme sparsity patterns than attention layers.

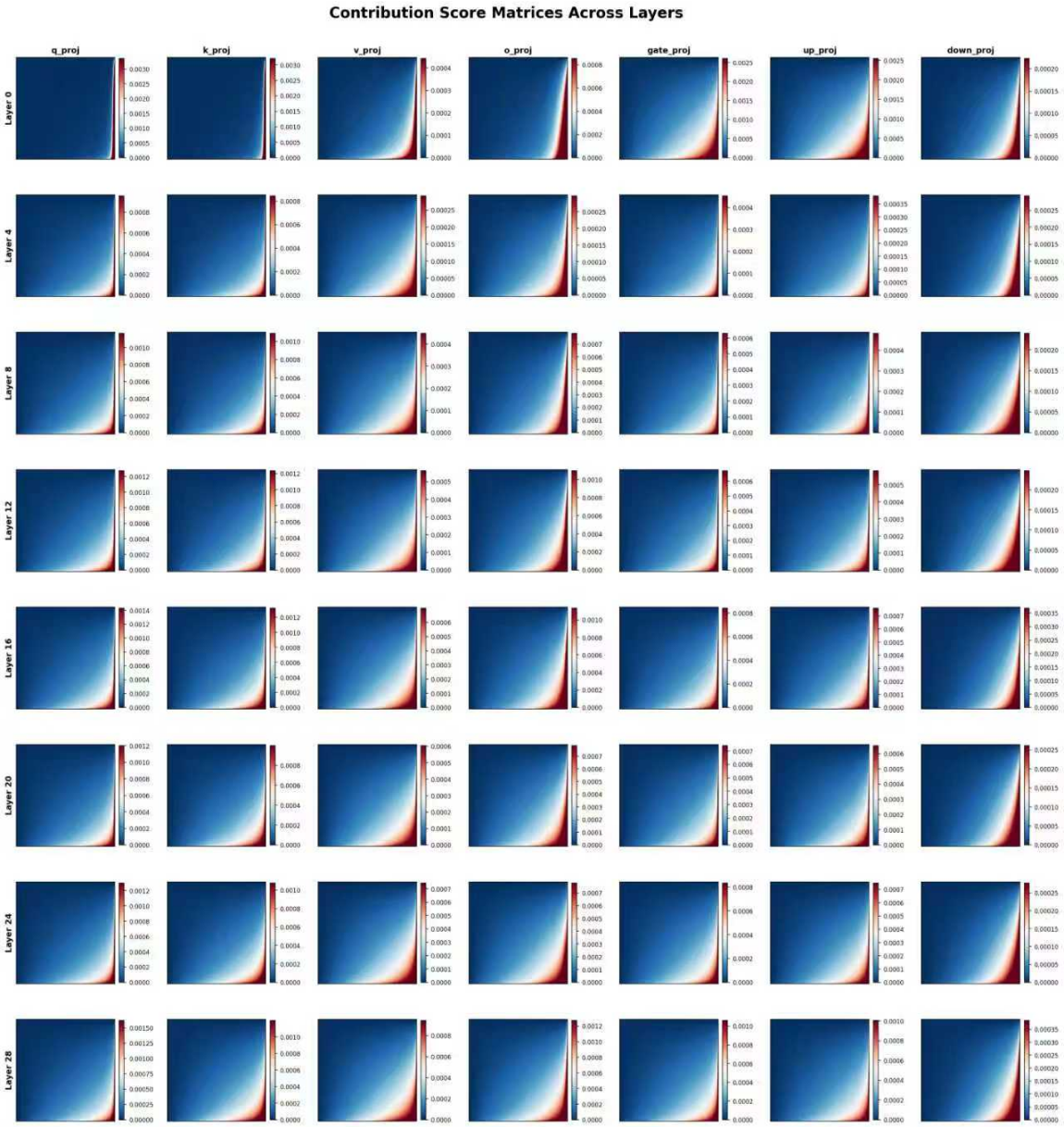

Figure 7: **Comprehensive layer-wise contribution analysis for Llama-2-7B.** Each subplot shows the cumulative contribution of activation-weight pairs for a specific layer and module type. Rows correspond to layers 0, 4, 8, 12, 16, 20, 24, 28 (sampled every 4 layers). Columns show different projection types: q_proj, k_proj, v_proj, o_proj (attention), gate_proj, up_proj, down_proj (FFN). The blue-to-red gradient indicates contribution magnitude, with red regions representing high contribution. Consistent heavy-tailed patterns across all modules validate our three-tier routing design. The steeper gradients in FFN projections (rightmost columns) confirm that FFN layers exhibit more extreme sparsity patterns than attention layers.

C Proofs of Theoretical Results

C.1 Proof of Theorem 3

Throughout, define the concentration function

$$\beta(\rho) := \frac{\|\mathbf{x}_{\text{top-}\rho}\|^2}{\|\mathbf{x}\|^2} \in [0,1],$$

which is non-decreasing in ρ and strictly concave under heavy-tailed activations. Assumption 1 corresponds to $\beta(\rho_h) \geq \beta_h$ and $\beta(\rho_h + \rho_m) \geq \beta_m$.

Part 1: Error Bound.

Proof. The error decomposes as

$$\begin{aligned}\mathbf{y}_{\text{DAWS}} - \mathbf{y}_{\text{dense}} &= \mathbf{W}\mathbf{x}_h + \mathbf{W}_p\mathbf{x}_m - \mathbf{W}(\mathbf{x}_h + \mathbf{x}_m + \mathbf{x}_l) \\ &= (\mathbf{W}_p - \mathbf{W})\mathbf{x}_m - \mathbf{W}\mathbf{x}_l. \end{aligned} \tag{12}$$

Applying $\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$ and submultiplicativity,

$$\begin{aligned}\|\mathbf{y}_{\text{DAWS}} - \mathbf{y}_{\text{dense}}\|^2 &\leq 2\|(\mathbf{W}_p - \mathbf{W})\mathbf{x}_m\|^2 + 2\|\mathbf{W}\mathbf{x}_l\|^2 \\ &\leq 2\epsilon^2(\alpha)\|\mathbf{W}\|_F^2\|\mathbf{x}_m\|^2 + 2\|\mathbf{W}\|_F^2\|\mathbf{x}_l\|^2 \qquad (13) \\ &\leq 2\big[\epsilon^2(\alpha)+1\big]\|\mathbf{W}\|_F^2\big[\|\mathbf{x}_m\|^2 + \|\mathbf{x}_l\|^2\big] \\ &\leq 2\big[\epsilon^2(\alpha)+1\big](1-\beta_h)\|\mathbf{W}\|_F^2\|\mathbf{x}\|^2, \qquad (14)\end{aligned}$$

where the second inequality uses Assumption 2, and the last uses $\|\mathbf{x}_m\|^2 + \|\mathbf{x}_l\|^2 = \|\mathbf{x}\|^2 - \|\mathbf{x}_h\|^2 \leq (1-\beta_h)\|\mathbf{x}\|^2$ from Assumption 1. □

Part 2: Superiority over Two-Tier Routing.

Proof. The two-tier scheme with activation ratio ρ produces output $\mathbf{y}_{\text{2-tier}} = \mathbf{W}\mathbf{x}_{\text{top-}\rho}$. Consider the three-tier scheme with parameters $(\rho_h, \rho_m, \alpha) = (\rho, 0, \alpha_0)$ for any $\alpha_0 \in [0,1]$. Then $\mathbf{x}_m = \mathbf{0}$, giving

$$\mathbf{y}_{\text{DAWS}}\big|_{\rho_m=0} = \mathbf{W}\mathbf{x}_h + \mathbf{W}_p \cdot \mathbf{0} = \mathbf{W}\mathbf{x}_{\text{top-}\rho} = \mathbf{y}_{\text{2-tier}},$$

with matching budget $\rho_h + \rho_m\alpha_0 = \rho$. Hence for any two-tier configuration with budget $\rho = S$, there exists a three-tier configuration with the same budget achieving identical error. Minimizing over the larger three-tier parameter space therefore yields

$$\min_{\rho_h + \rho_m\alpha = S} \|\mathbf{y}_{\text{DAWS}} - \mathbf{y}_{\text{dense}}\|^2 \;\leq\; \min_{\rho = S} \|\mathbf{y}_{\text{2-tier}} - \mathbf{y}_{\text{dense}}\|^2.$$

Strict inequality holds whenever some (ρ_h, ρ_m, α) with $\rho_m > 0$ and $\alpha < 1$ outperforms all $\rho_m = 0$ configurations. Under heavy-tailed activations this regime is empirically dominant. □

C.2 Proof of Proposition 4

Proof. We decompose $\mathbf{W}_p^{(j)} - \mathbf{W}$ around the sample-averaged mask $\bar{\mathbf{W}}_p$:

$$\mathbf{W}_p^{(j)} - \mathbf{W} = \underbrace{(\bar{\mathbf{W}}_p - \mathbf{W})}_{\text{independent of } j} + \underbrace{(\mathbf{W}_p^{(j)} - \bar{\mathbf{W}}_p)}_{\text{zero-mean in } j},$$

where the second term has zero mean since $\mathbb{E}_j[\mathbf{W}_p^{(j)} - \bar{\mathbf{W}}_p] = \bar{\mathbf{W}}_p - \bar{\mathbf{W}}_p = \mathbf{0}$ by the definition of $\bar{\mathbf{W}}_p$.

Applying this decomposition to $\mathbf{x}_m^{(i)}$ and using the identity $\|\mathbf{a}+\mathbf{b}\|^2 = \|\mathbf{a}\|^2 + 2\langle \mathbf{a}, \mathbf{b}\rangle + \|\mathbf{b}\|^2$,

$$\begin{aligned}&\|(\mathbf{W}_p^{(j)} - \mathbf{W})\mathbf{x}_m^{(i)}\|^2 \\ &= \|(\bar{\mathbf{W}}_p - \mathbf{W})\mathbf{x}_m^{(i)}\|^2 + \|(\mathbf{W}_p^{(j)} - \bar{\mathbf{W}}_p)\mathbf{x}_m^{(i)}\|^2 \\ &\quad + 2\big\langle (\bar{\mathbf{W}}_p - \mathbf{W})\mathbf{x}_m^{(i)},\, (\mathbf{W}_p^{(j)} - \bar{\mathbf{W}}_p)\mathbf{x}_m^{(i)} \big\rangle. \end{aligned} \tag{15}$$

Now take $\mathbb{E}_{j,i}$ on both sides of (15) term by term.

Term 1 (bias). The integrand is independent of j, so

$$\mathbb{E}_{j,i}\|(\bar{\mathbf{W}}_p - \mathbf{W})\mathbf{x}_m^{(i)}\|^2 = \mathbb{E}_i\|(\bar{\mathbf{W}}_p - \mathbf{W})\mathbf{x}_m^{(i)}\|^2.$$

Term 2 (variance). This term stays as is:

$$\mathbb{E}_{j,i}\|(\mathbf{W}_p^{(j)} - \bar{\mathbf{W}}_p)\mathbf{x}_m^{(i)}\|^2.$$

Term 3 (cross term). Using $j \perp i$, we can apply Fubini's theorem and take the expectation over j inside the inner product; moreover, $(\bar{\mathbf{W}}_p - \mathbf{W})\mathbf{x}_m^{(i)}$ does not depend on j, so it factors out of $\mathbb{E}_j$:

$$\begin{aligned}
&\mathbb{E}_{j,i}\Big[2\big\langle(\bar{\mathbf{W}}_p - \mathbf{W})\mathbf{x}_m^{(i)},\, (\mathbf{W}_p^{(j)} - \bar{\mathbf{W}}_p)\mathbf{x}_m^{(i)}\big\rangle\Big] \\
&= 2\,\mathbb{E}_i\Big[\big\langle(\bar{\mathbf{W}}_p - \mathbf{W})\mathbf{x}_m^{(i)},\, \mathbb{E}_j[\mathbf{W}_p^{(j)} - \bar{\mathbf{W}}_p]\,\mathbf{x}_m^{(i)}\big\rangle\Big] \\
&= 2\,\mathbb{E}_i\Big[\big\langle(\bar{\mathbf{W}}_p - \mathbf{W})\mathbf{x}_m^{(i)},\, \mathbf{0}\big\rangle\Big] = 0.
\end{aligned}$$

Combining the three terms gives

$$\begin{aligned}
&\mathbb{E}_{j,i}\|(\mathbf{W}_p^{(j)} - \mathbf{W})\mathbf{x}_m^{(i)}\|^2 \\
&= \mathbb{E}_i\|(\bar{\mathbf{W}}_p - \mathbf{W})\mathbf{x}_m^{(i)}\|^2 \\
&\quad + \mathbb{E}_{j,i}\|(\mathbf{W}_p^{(j)} - \bar{\mathbf{W}}_p)\mathbf{x}_m^{(i)}\|^2,
\end{aligned}$$

which is the claimed decomposition.

The variance term is a squared norm, hence non-negative, and equals zero iff $(\mathbf{W}_p^{(j)} - \bar{\mathbf{W}}_p)\mathbf{x}_m^{(i)} = \mathbf{0}$ almost surely for all (j, i). Since the test input $\mathbf{x}_m^{(i)}$ has generic support, this holds iff $\mathbf{W}_p^{(j)} = \bar{\mathbf{W}}_p$ for all j, i.e., iff $\mathbf{W}_p^{(j)}$ is constant in j. □

D AI Assistance Disclosure

In preparing this manuscript, the authors used AI language models (Claude 4.5 Sonnet, Anthropic) to assist with language refinement and manuscript organization. Specifically, AI tools were used to improve the clarity and grammatical correctness of technical descriptions, suggest structural improvements for better readability, and refine figure and table captions to ensure they are self-contained. All core scientific contributions, including the proposed DAWS method, theoretical analysis, experimental design, implementation, and interpretation of results, were conceived and developed entirely by the authors without AI assistance. The authors have carefully reviewed all AI-assisted content for accuracy and take full responsibility for the entire manuscript, including any errors or omissions.