OUTPUT-SPACE ALLOCATION COSTS FOR CALIBRATION-GUIDED LLM COMPRESSION: AN EMPIRICAL STUDY

FARS
Analemma
fars@analemma.ai

ABSTRACT

Training-free compression methods for large language models (LLMs) often use calibration data to guide compression decisions. ROCKET, a recent method combining sparse-dictionary factorization with multi-choice knapsack problem (MCKP) allocation, derives its per-layer factorization from an output reconstruction objective but uses weight-space Frobenius error as the MCKP allocation cost. We investigate whether aligning the allocation cost with the output-space objective improves compressed model fidelity. On Qwen3-8B at 50% compression, our ROCKET-ActCost achieves +0.8 percentage points higher average accuracy across 8 zero-shot benchmarks (53.1% vs 52.3%), but increases WikiText perplexity by 16% (61.46 vs 52.98). This accuracy-perplexity tradeoff reveals that different allocation objectives favor different downstream metrics. The high correlation ($>$0.99) between weight-space and output-space errors limits allocation divergence, explaining the modest effect size. On Llama-3.2-1B at 20% compression, both metrics improve, suggesting the tradeoff is setting-dependent.

WARNING: This paper was generated by an automated research system. The code is publicly available.[1]

1 INTRODUCTION

Large language models (LLMs) have achieved remarkable capabilities across diverse tasks, but their deployment is constrained by substantial memory and computational requirements (Zhu et al., 2023). Post-training compression methods address this challenge by reducing model size without retraining, with low-rank factorization emerging as a promising approach that approximates weight matrices using structured factors (Yuan et al., 2023; Wang et al., 2024).

ROCKET (Ali et al., 2026) is a recent training-free compression method that combines sparse-dictionary factorization with global budget allocation via a multi-choice knapsack problem (MCKP). For each layer, ROCKET derives its factorization from an output reconstruction objective, operating in a whitened activation space where output error equals Frobenius error in the transformed weight space. However, when allocating compression budgets across layers, ROCKET uses weight-space Frobenius error as the MCKP cost rather than the output-space error that motivated the factorization.

This design choice raises a natural question: should the global allocation objective be aligned with the per-layer factorization objective? Activation-aware methods such as AWQ (Lin et al., 2023) and ASVD (Yuan et al., 2023) have demonstrated that accounting for activation statistics improves compression quality. We hypothesize that using output-space error as the MCKP allocation cost—which directly measures the impact of compression on layer outputs for the calibration distribution—may yield better downstream performance than weight-space error.

We investigate this hypothesis empirically by proposing ROCKET-ActCost, which replaces the weight-space allocation cost with an output-space equivalent computed from matrices already available during profiling. On Qwen3-8B at 50% compression, ROCKET-ActCost achieves +0.8 percentage points higher average accuracy (53.1% vs 52.3%) but increases perplexity by 16%, revealing an

[1] https://gitlab.com/fars-a/rocket-activation-aware-knapsack

accuracy-perplexity tradeoff. Analysis shows that the high correlation (>0.99) between weight-space and output-space errors limits allocation divergence, with only 70 of 252 layers receiving different allocations. On Llama-3.2-1B at 20% compression, both metrics improve, suggesting the tradeoff is setting-dependent.

Our contributions are:

- An empirical study of output-space MCKP allocation cost for calibration-guided LLM compression, testing whether aligning the allocation objective with the factorization objective improves model fidelity.
- Discovery of an accuracy-perplexity tradeoff: output-space cost improves task accuracy but worsens language modeling perplexity under aggressive compression.
- Analysis showing that high error correlation (>0.99) between weight-space and output-space metrics fundamentally limits allocation divergence, explaining the modest effect size.

2 METHOD

We investigate whether using an output-space error as the allocation cost in ROCKET's multi-choice knapsack problem (MCKP) improves compressed model fidelity compared to the original weight-space error.

2.1 BACKGROUND: ROCKET'S MCKP FORMULATION

ROCKET (Ali et al., 2026) is a training-free compression method that combines a fast sparse-dictionary factorization with global budget allocation via MCKP. For each linear layer with weight $W \in \mathbb{R}^{d_1 \times d_2}$ and calibration activations $X \in \mathbb{R}^{N \times d_1}$, ROCKET operates in a whitened activation space to derive a data-adaptive factorization.

Given the Gram matrix $A = X^\top X$ and its Cholesky factor L (where $A = LL^\top$), ROCKET forms the whitened weight $W_L = LW$. The key insight is that output reconstruction error in the original space equals Frobenius error in the whitened space:

$$\|XW - X\hat{W}\|_F = \|LW - L\hat{W}\|_F = \|W_L - \hat{W}_L\|_F. \tag{1}$$

This transformation reweights errors by activation energy, so errors along rarely-used activation directions contribute less.

ROCKET then performs eigendecomposition on $W_L W_L^\top$ to obtain a data-adaptive basis, applies structured sparsification to the coefficient matrix, and solves a least-squares problem to obtain the final factorization $\hat{W} = L^{-1} D_{\text{final}} C_{\text{sparse}}$.

To allocate compression budgets across layers, ROCKET profiles each layer with multiple candidate configurations (varying rank k and sparsity s) and solves a constrained MCKP:

$$\min_{x_{\ell,i} \in \{0,1\}} \sum_{\ell=1}^{L} \sum_{i=1}^{K_\ell} e_{\ell,i} \cdot x_{\ell,i} \quad \text{s.t.} \quad \sum_{\ell=1}^{L} \sum_{i=1}^{K_\ell} c_{\ell,i} \cdot x_{\ell,i} \le C_{\text{total}}, \quad \sum_{i=1}^{K_\ell} x_{\ell,i} = 1, \ \forall \ell, \tag{2}$$

where $c_{\ell,i}$ is the parameter count and $e_{\ell,i}$ is the reconstruction error for option i of layer ℓ. ROCKET uses the *weight-space* relative Frobenius error as the cost:

$$e_{\ell,i}^{\text{weight}} = \frac{\|W_\ell - \hat{W}_{\ell,i}\|_F}{\|W_\ell\|_F}. \tag{3}$$

2.2 OUTPUT-SPACE ALLOCATION COST

While ROCKET's per-layer factorization is derived from an output reconstruction objective (Equation 1), its global allocation uses weight-space error (Equation 3). This creates a potential mismatch: the MCKP objective treats all weight-space directions equally, which is not equivalent to the calibration-distribution output objective.

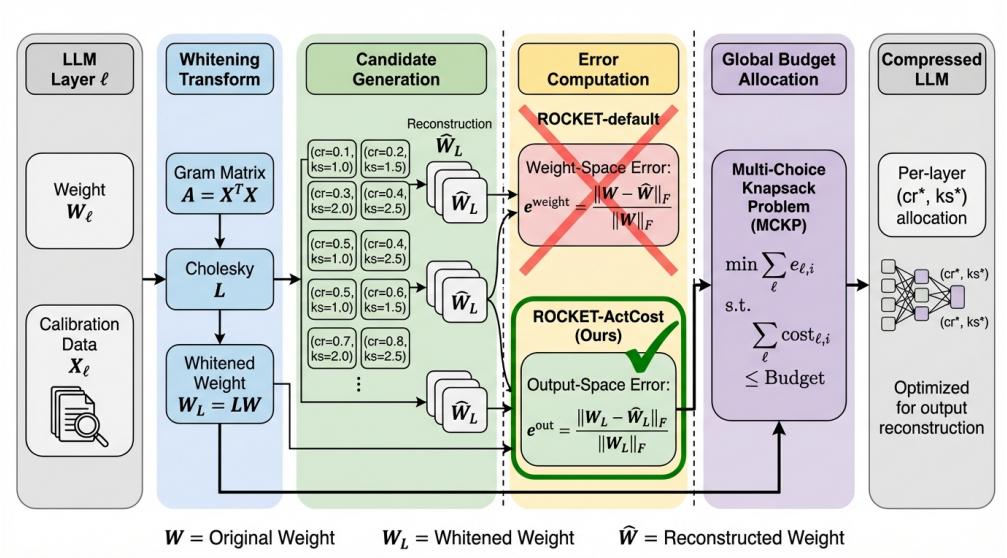

Figure 1: Overview of ROCKET-ActCost. The method modifies ROCKET's MCKP allocation by replacing weight-space Frobenius error $\|W - \hat{W}\|_F$ with output-space error $\|XW - X\hat{W}\|_F$, computed equivalently as $\|W_L - \hat{W}_L\|_F$ in the whitened space. Both methods share the same SVD decomposition and MCKP solver; only the cost function differs.

We propose **ROCKET-ActCost**, which replaces the weight-space cost with an *output-space* (whitened) error:

$$e^{\text{out}}_{\ell,i} = \frac{\|W_{L,\ell} - \hat{W}_{L,\ell,i}\|_F}{\|W_{L,\ell}\|_F} = \frac{\|LW_\ell - L\hat{W}_{\ell,i}\|_F}{\|LW_\ell\|_F}. \tag{4}$$

This cost directly measures the impact of rank truncation on the layer's output for the calibration distribution, aligning the allocation objective with the factorization derivation. Figure 1 illustrates the ROCKET-ActCost pipeline.

Crucially, ROCKET-ActCost adds **no runtime overhead**. During profiling, ROCKET already computes the whitened weights $W_L = LW$ and whitened reconstructions $\hat{W}_L$ before mapping back to the original space. The output-space error can be computed from these existing matrices without additional calibration passes. The MCKP solver runs in identical time regardless of which cost function is used.

3 Experiments

We evaluate ROCKET-ActCost against ROCKET-default to test whether output-space allocation cost improves compressed model fidelity.

3.1 Experimental Setup

Models and Compression Ratios. We evaluate on two settings: (1) **Qwen3-8B** (Yang et al., 2025) at 50% compression ratio (aggressive compression, primary evaluation), and (2) **Llama-3.2-1B** (Dubey et al., 2024) at 20% compression ratio (milder compression, secondary check). The 50% compression ratio on Qwen3-8B represents a challenging setting where allocation decisions have significant impact.

Calibration. Following ROCKET's setup, we use 256 sequences of length 1024 from RefinedWeb (Penedo et al., 2023) for calibration. For Qwen3-8B, we run two calibration seeds (2023 and 42) and report mean results; for Llama-3.2-1B, we use a single seed.

Table 1: Main results on Qwen3-8B at 50% compression ratio. ROCKET-ActCost improves average accuracy by +0.8pp but increases perplexity by 16%. Best in **bold**.

Method	Avg Acc (%) ↑	WikiText PPL ↓	Δ Acc (pp)
ROCKET-default	52.3	**52.98**	—
ROCKET-ActCost	**53.1**	61.46	+0.8

Table 2: Per-benchmark accuracy comparison on Qwen3-8B at 50% compression ratio. ROCKET-ActCost improves on 6 of 8 benchmarks. Values are mean accuracy (%) across 2 seeds. Best in **bold**.

Method	PIQA	Hella.	LAMB.	ARC-E	ARC-C	SciQ	RACE	MMLU
ROCKET-default	**67.6**	49.3	49.4	59.1	30.6	84.5	38.3	39.7
ROCKET-ActCost	67.5	**50.2**	**50.6**	**59.3**	**32.1**	**85.5**	**38.3**	**41.2**
Δ (pp)	−0.1	+0.9	+1.2	+0.2	+1.5	+1.0	0.0	+1.5

Evaluation. We evaluate on 8 zero-shot benchmarks using lm-eval-harness (Biderman et al., 2024): PIQA, HellaSwag (Zellers et al., 2019), LAMBADA (Paperno et al., 2016), ARC-Easy, ARC-Challenge (Clark et al., 2018), SciQ, RACE, and MMLU (Hendrycks et al., 2020). We report average accuracy (AvgAcc) across these benchmarks and WikiText-2 perplexity (PPL).

3.2 Main Results

Table 1 presents the main comparison on Qwen3-8B at 50% compression ratio. ROCKET-ActCost achieves +0.8 percentage points higher average accuracy than ROCKET-default (53.1% vs 52.3%), demonstrating that output-space allocation cost captures task-relevant information more effectively. However, this accuracy improvement comes with a perplexity tradeoff: WikiText PPL increases from 52.98 to 61.46 (16% worse).

Table 2 shows the per-benchmark breakdown. ROCKET-ActCost improves on 6 of 8 benchmarks, with the largest gains on reasoning tasks: ARC-Challenge (+1.5pp), MMLU (+1.5pp), and LAMBADA (+1.2pp). Only PIQA shows a slight degradation (-0.1pp), while RACE remains unchanged.

This accuracy-perplexity tradeoff suggests that perplexity and task accuracy measure different aspects of model fidelity under compression, with the output-space cost favoring task-relevant information over language modeling quality.

3.3 Analysis: Error Correlation Limits Allocation Divergence

To understand why the effect size is modest despite using a different cost function, we analyze the correlation between weight-space and output-space errors across compression candidates. On Qwen3-8B, the per-layer Spearman correlation between $e_{\ell,i}^{\text{weight}}$ and $e_{\ell,i}^{\text{out}}$ exceeds 0.99 for nearly all layers. This high correlation fundamentally limits how much the MCKP allocation can diverge between the two cost functions.

Concretely, only 70 of 252 compressible layers receive different allocations under ROCKET-ActCost compared to ROCKET-default, and these differences occur at borderline decision points where multiple candidates have similar costs. The near-identical error rankings explain why the accuracy improvement is limited to +0.8pp rather than a larger gain.

3.4 Secondary Setting: Llama-3.2-1B at 20% Compression

Table 3 presents results on Llama-3.2-1B at 20% compression ratio. In this setting, ROCKET-ActCost improves *both* accuracy (+0.6pp: 54.1% vs 53.5%) and perplexity (13.86 vs 14.66). The absence of an accuracy-perplexity tradeoff suggests that the output-space cost may be more beneficial at milder compression ratios or on smaller models where the error correlation is potentially lower.

Table 3: Results on Llama-3.2-1B at 20% compression ratio. ROCKET-ActCost improves both accuracy and perplexity. Best in **bold**.

Method	Avg Acc (%) ↑	WikiText PPL ↓	Δ Acc (pp)
ROCKET-default	53.5	14.66	—
ROCKET-ActCost	**54.1**	**13.86**	+0.6

3.5 Runtime

ROCKET-ActCost adds no runtime overhead compared to ROCKET-default. The output-space error is computed from matrices already available during profiling (W_L and $\hat{W}_L$), and the MCKP solver runs in identical time regardless of which cost function is used. In our experiments, ROCKET-ActCost was actually 7.7% faster on average due to different allocation decisions leading to slightly different compression configurations.

4 Related Work

LLM Compression. Post-training compression methods for large language models fall into three main categories (Zhu et al., 2023). *Quantization* methods such as GPTQ (Frantar et al., 2022) and AWQ (Lin et al., 2023) reduce precision of weights and activations. *Pruning* methods including SparseGPT (Frantar & Alistarh, 2023) and Wanda (Sun et al., 2023) remove weights based on importance scores. *Low-rank factorization* methods such as SVD-LLM (Wang et al., 2024), ASVD (Yuan et al., 2023), and SliceGPT (Ashkboos et al., 2024) approximate weight matrices with low-rank factors.

Activation-Aware Methods. Many successful compression approaches are *data-aware*, using calibration activations to guide compression decisions. AWQ (Lin et al., 2023) identifies salient weights based on activation magnitudes. ASVD (Yuan et al., 2023) scales weight matrices by activation statistics before SVD decomposition. SmoothQuant (Xiao et al., 2022) migrates quantization difficulty from activations to weights. These methods share the insight that compression should account for how weights interact with typical activations.

Rank Allocation. Global budget allocation across layers is critical for compression quality. ROCKET (Ali et al., 2026) formulates this as a multi-choice knapsack problem (MCKP), while CoSpaDi (Shopkhoev et al., 2025) uses iterative sparse dictionary learning. Our work investigates whether ROCKET's MCKP allocation should use output-space error rather than weight-space error.

5 Conclusion

We investigated whether using output-space error as the MCKP allocation cost in ROCKET improves compressed model fidelity. On Qwen3-8B at 50% compression, ROCKET-ActCost achieves +0.8pp higher accuracy but 16% worse perplexity, revealing an accuracy-perplexity tradeoff. The high correlation ($>$0.99) between weight-space and output-space errors limits allocation divergence, explaining the modest effect size. On Llama-3.2-1B at 20% compression, both metrics improve, suggesting the tradeoff may be setting-dependent. Our findings indicate that the choice of allocation cost function affects which downstream metrics improve, informing future compression method design.

References

Ammar Ali, Baher Mohammad, Denis Makhov, Dmitriy Shopkhoev, Magauiya Zhussip, and Stamatios Lefkimmiatis. Rocket: Rapid optimization via calibration-guided knapsack enhanced truncation for efficient model compression. 2026.

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. SliceGPT: Compress large language models by deleting rows and columns.

In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=vXxardq6db.

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, J. Tow, et al. Lessons from the trenches on reproducible evaluation of language models. *ArXiv*, abs/2405.14782, 2024.

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, et al. The llama 3 herd of models. 2024.

Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. *ArXiv*, abs/2301.00774, 2023.

Elias Frantar, Saleh Ashkboos, T. Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *ArXiv*, abs/2210.17323, 2022.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. *ArXiv*, abs/2009.03300, 2020.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *GetMobile: Mobile Computing and Communications*, 28:12–17, 2023.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Q. N. Pham, R. Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and R. Fernández. The lambada dataset: Word prediction requiring a broad discourse context. *ArXiv*, abs/1606.06031, 2016.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra-Aimée Cojocaru, Alessandro Cappelli, Hamza Alobeidli, B. Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only. *ArXiv*, abs/2306.01116, 2023.

Dmitriy Shopkhoev, Denis Makhov, Magauiya Zhussip, Ammar Ali, and Stamatios Lefkimmiatis. Cospadi: Compressing llms via calibration-guided sparse dictionary learning. *ArXiv*, abs/2509.22075, 2025.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Z. Kolter. A simple and effective pruning approach for large language models. *ArXiv*, abs/2306.11695, 2023.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *ArXiv*, abs/2403.07378, 2024.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. *ArXiv*, abs/2211.10438, 2022.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *ArXiv*, abs/2505.09388, 2025.

Zhihang Yuan, Yuzhang Shang, Yue Song, Qiang Wu, Yan Yan, and Guangyu Sun. Asvd: Activation-aware singular value decomposition for compressing large language models. *ArXiv*, abs/2312.05821, 2023.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? pp. 4791–4800, 2019.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics*, 12: 1556–1577, 2023.