MORS: METRIC-ORTHOGONAL SPARSE FACTORIZATION FOR LOSS-AWARE LLM COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse dictionary learning has shown advantages over SVD-based compression by providing a more flexible weight representation, yet loss-aware formulations have remained largely confined to low-rank methods. We introduce **MORS**, a loss-aware sparse dictionary learning framework that incorporates second-order information from a Kronecker-factored empirical Fisher approximation into both the factorization objective and compression allocation. MORS optimizes the factorization directly in the original weight domain while constraining the dictionary to be orthogonal under the activation-side curvature metric. The general formulation supports full output-side curvature, while a diagonal surrogate decouples sparse support selection and admits efficient closed-form updates. Across diverse model families and compression ratios, MORS consistently outperforms strong SVD-based baselines, establishing loss-aware sparse dictionary compression as a competitive alternative to low-rank factorization.

1 INTRODUCTION

Low-rank matrix factorization has become a predominant paradigm for structured post-training compression of large language models (LLMs). Most approaches build on singular value decomposition (SVD), replacing large projection matrices with compact low-rank factors and progressively improving the approximation using calibration data, activation statistics, and non-uniform rank allocation (Chen et al., 2021; Yuan et al., 2023; Wang et al., 2025b;a). Despite its effectiveness, a rank-r approximation necessarily represents all reconstructed columns within the same r-dimensional subspace, which can become restrictive at high compression ratios.

Sparse dictionary learning provides a more flexible alternative. Instead of constraining the entire matrix to a single low-dimensional subspace, it allows each weight column to select a sparse subset of atoms from a larger dictionary, effectively forming a union of subspaces. CoSpaDi demonstrated the potential of this representation for post-training LLM compression (Makhov et al., 2026a), while COMPOT showed that imposing dictionary orthogonality enables efficient hard-thresholding and Procrustes updates and further improves the compression–quality trade-off (Makhov et al., 2026b). These results establish sparse dictionary learning as a competitive alternative to conventional low-rank factorization.

However, both low-rank and dictionary-based post-training methods have predominantly been designed to reproduce the pretrained weights or layer activations. Reconstruction quality is only a proxy for the ultimate objective: preserving the model loss. Recent loss-aware SVD methods address this mismatch by incorporating Fisher or curvature information into low-rank decomposition and rank allocation (Hsu et al., 2022; Chekalina et al., 2026; Thoma et al., 2026; Wang et al., 2026). This raises a natural question: can the same loss-aware principle be transferred to the more expressive sparse dictionary representation?

We answer this question with **MORS** (**M**etric-**OR**thogonal **S**parse Factorization), a loss-aware sparse dictionary learning framework based on a Kronecker-factored approximation of the empirical Fisher. MORS brings second-order loss sensitivity into the sparse factorization while preserving the efficient alternating structure of orthogonal dictionary learning. We also revisit dynamic compression allocation. Existing dictionary-based allocation relies on an SVD-derived spectral proxy despite ultimately deploying a sparse dictionary representation (Makhov et al., 2026b). MORS in-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

stead evaluates candidate sparse representations directly under the loss-aware metric, aligning the criterion used for global allocation with the representation used for compression.

Overall, the contributions of this work are threefold: **(a)** we introduce a computationally efficient *loss-aware sparse dictionary compression* strategy, which combines the representational flexibility of sparse dictionaries with second-order loss information derived from a Kronecker-factored empirical Fisher approximation, together with an efficient alternating optimization procedure; **(b)** we propose a *representation-aware dynamic allocation strategy* that evaluates candidate sparse factorizations according to their predicted impact on the model loss and allocates the compression budget accordingly; and **(c)** through extensive experiments across model families and compression ratios, we demonstrate that MORS consistently improves compression quality not only for text-only LLMs, but also for vision–language and speech models.

2 Related Work

SVD-based model compression. Low-rank compression has progressively moved from weight reconstruction toward objectives that better reflect model behavior. Standard truncated SVD provides the optimal rank-constrained approximation under the Frobenius norm, while data-aware methods incorporate calibration activations. DRONE directly minimizes linear-layer output reconstruction under a rank constraint (Chen et al., 2021); ASVD rescales weight channels according to activation statistics (Yuan et al., 2023); and SVD-LLM uses activation second moments to derive a whitening transformation whose truncation error corresponds to layer-output reconstruction (Wang et al., 2025b). SVD-LLM V2 further improves the truncation criterion and introduces non-uniform compression ratios (Wang et al., 2025a). A complementary line explicitly incorporates task-loss sensitivity. FWSVD weights approximation errors using empirical Fisher information (Hsu et al., 2022), while recent GFWSVD, KFAC-SVD, and LASER use richer Fisher or Kronecker-factored curvature approximations for loss-aware decomposition (Chekalina et al., 2026; Thoma et al., 2026; Wang et al., 2026). These methods progressively improve the metric used to construct a low-rank approximation, but all remain constrained to a rank-r representation in which the reconstructed columns share the column space of the left factor.

Sparse dictionary compression. Sparse dictionary learning provides a more flexible representation by expressing each weight vector using only a subset of atoms from a learned dictionary. Classical methods such as K-SVD alternate sparse coding with dictionary updates (Aharon et al., 2006), while orthogonal dictionary learning shows that imposing additional structure can substantially simplify optimization (Bao et al., 2013). CoSpaDi applies sparse dictionary learning directly to post-training LLM compression and uses calibration activations to optimize layer-output reconstruction, allowing different weight columns to select different subsets of dictionary atoms (Makhov et al., 2026a). COMPOT further imposes orthogonality in the activation-aware space, reducing sparse coding to hard thresholding and the dictionary update to an orthogonal Procrustes problem (Makhov et al., 2026b). Thus, sparse dictionary methods have followed the same transition from weight-only to activation-aware reconstruction as low-rank compression, but have not yet incorporated the richer loss-aware objectives recently developed for SVD. MORS closes this gap by optimizing sparse dictionary factors under a second-order loss metric while retaining efficient metric-orthogonal updates.

Dynamic compression allocation. A separate challenge is distributing a global compression budget across heterogeneous projection matrices. Early approaches estimate layer sensitivity directly, as in ASVD's calibration-based rank search (Yuan et al., 2023), while later methods increasingly derive allocation criteria from the compressed representation or loss geometry. SVD-LLM V2 uses a truncation-loss model and LEMS fits a calibrated layer-wise error surrogate for global rank allocation (Wang et al., 2025a; Thoma et al., 2026). LASER derives allocation from curvature information (Wang et al., 2026), Swift-SVD combines local reconstruction error with end-to-end layer importance (Qi et al., 2026), ZS-SVD globally selects whitened singular components using first-order loss estimates (Abbasi et al., 2026), and LACE-SVD additionally models cumulative compression-error propagation through the residual stream (Liu et al., 2026). For sparse dictionary compression, COMPOT allocates layer-wise budgets from the singular-value spectra of the original weights (Makhov et al., 2026b). In contrast, MORS directly scores candidate sparse factorizations under the same loss-aware metric used by the local compressor, aligning global allocation with both

the deployed representation and its optimization objective, while not explicitly modeling cross-layer perturbation interactions.

3 METHOD

3.1 LOSS-AWARE COMPRESSION OBJECTIVE

Consider a linear projection with weight matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$, input activations $\mathbf{X} \in \mathbb{R}^{N\times m}$, and output $\mathbf{Y} = \mathbf{X}\mathbf{W} \in \mathbb{R}^{N\times n}$. We replace $\mathbf{W}$ with a compressed approximation $\widehat{\mathbf{W}}$ and denote the induced perturbation by $\mathbf{E} = \widehat{\mathbf{W}} - \mathbf{W}$. Let $\mathcal{L}$ denote the calibration loss, viewed as a function of the projection weights $\mathbf{W}$ with all other model parameters fixed, and define $\Delta\mathcal{L} = \mathcal{L}(\widehat{\mathbf{W}}) - \mathcal{L}(\mathbf{W})$. A second-order Taylor expansion around the pretrained parameters gives

$$\Delta\mathcal{L} \approx \langle \nabla_{\mathbf{W}}\mathcal{L}, \mathbf{E}\rangle_F + \frac{1}{2}\operatorname{vec}(\mathbf{E})^\top \mathbf{H}_{\mathbf{W}} \operatorname{vec}(\mathbf{E}), \tag{1}$$

where $\mathbf{H}_{\mathbf{W}}$ is the curvature with respect to the projection weights. In this work we focus on the second-order contribution and omit the first-order term, treating the pretrained model as locally close to a stationary point. Retaining the first-order term preserves the quadratic structure of the objective and only shifts its effective target weights (see Appendix A.1).

Direct construction of $\mathbf{H}_{\mathbf{W}}$ is infeasible for transformer projection matrices. We therefore replace the Hessian with the empirical Fisher matrix, which provides a positive-semidefinite curvature surrogate. Let $\mathbf{G} = \partial\mathcal{L}/\partial\mathbf{Y} \in \mathbb{R}^{N\times n}$ denote the output gradients collected on the calibration set. For an individual token with input $\mathbf{x}$ and output gradient $\mathbf{g}$, the gradient with respect to the projection weights is $\mathbf{x}\mathbf{g}^\top$. The empirical Fisher block for $\mathbf{W}$ is therefore

$$\mathbf{F}_{\mathbf{W}} = \mathbb{E}\left[\operatorname{vec}(\mathbf{x}\mathbf{g}^\top)\operatorname{vec}(\mathbf{x}\mathbf{g}^\top)^\top\right]. \tag{2}$$

Although substantially more structured than the full Hessian, $\mathbf{F}_{\mathbf{W}}$ is still prohibitively large to construct explicitly. We therefore apply the Kronecker-Factored Approximate Curvature (KFAC) approximation (Martens & Grosse, 2015), which factorizes the empirical Fisher as

$$\mathbf{F}_{\mathbf{W}} \approx \mathbf{B} \otimes \mathbf{A}, \qquad \mathbf{A} = \mathbb{E}[\mathbf{x}\mathbf{x}^\top], \qquad \mathbf{B} = \mathbb{E}[\mathbf{g}\mathbf{g}^\top]. \tag{3}$$

Here, $\mathbf{A}$ captures correlations between the projection inputs, while $\mathbf{B}$ captures correlations between its output gradients. Because these empirical second-moment estimates are computed from a finite calibration set and may therefore be rank-deficient or poorly conditioned, we consider the damped matrices (see Appendix A) $\mathbf{A}_\lambda = \mathbf{A} + \lambda_x\mathbf{I}$, $\mathbf{B}_\lambda = \mathbf{B} + \lambda_g\mathbf{I}$, and obtain the tractable second-order compression surrogate

$$\mathcal{Q}(\mathbf{E}) = \frac{1}{2}\operatorname{tr}\left(\mathbf{E}^\top\mathbf{A}_\lambda\mathbf{E}\mathbf{B}_\lambda\right). \tag{4}$$

In this setting, each projection perturbation is evaluated independently around the pretrained model, corresponding to a block-diagonal approximation of the model-wide curvature that neglects cross-projection interactions. Setting $\mathbf{B}_\lambda = \mathbf{I}$ removes output-side curvature weighting, reducing the objective to an activation-aware approximation governed solely by the input-side metric $\mathbf{A}_\lambda$.

3.2 METRIC-ORTHOGONAL SPARSE DICTIONARY LEARNING

MORS represents a compressed projection as $\widehat{\mathbf{W}} = \mathbf{D}\mathbf{S}$, where $\mathbf{D} \in \mathbb{R}^{m\times k}$ is a dense dictionary and $\mathbf{S} \in \mathbb{R}^{k\times n}$ is a sparse coefficient matrix. We consider complete or undercomplete dictionaries ($k \le m$), impose orthogonality under the activation-side curvature metric, $\mathbf{D}^\top\mathbf{A}_\lambda\mathbf{D} = \mathbf{I}_k$, and constrain each coefficient column by $\|\mathbf{s}_j\|_0 \le s$. We follow COMPOT (Makhov et al., 2026b) for the dictionary-to-sparsity parameterization and storage accounting. For an $m \times n$ projection represented by a dense dictionary $\mathbf{D} \in \mathbb{R}^{m\times k}$ and s non-zero coefficients per output column, the FP16-equivalent storage is

$$P_{\text{MORS}} = mk + sn + \frac{kn}{16}, \tag{5}$$

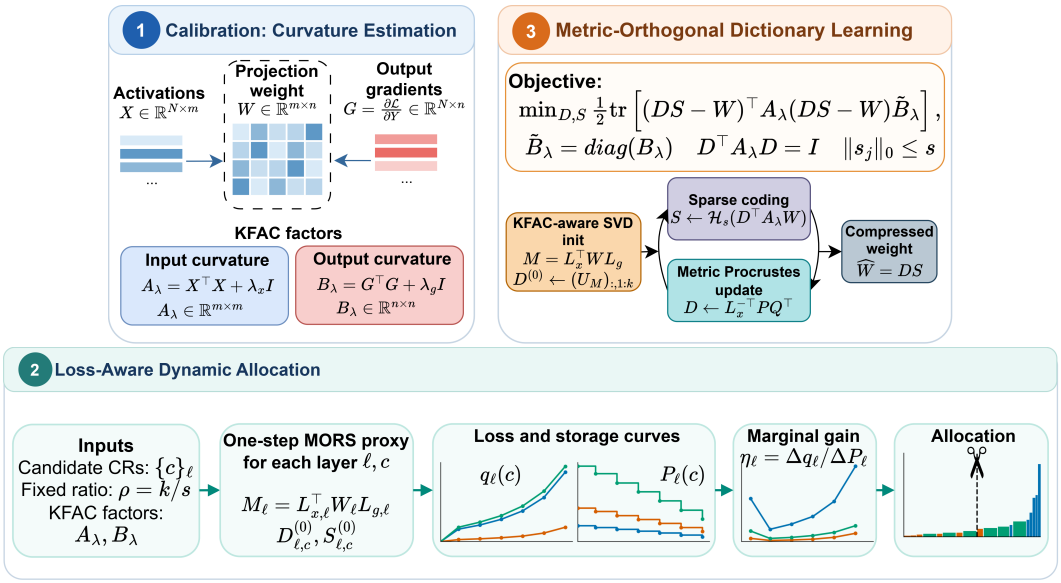

Figure 1: Overview of MORS. Calibration provides the KFAC statistics used by both the metric-orthogonal sparse factorization and the loss-aware dynamic allocation.

where the last term accounts for the binary support mask. We define the compression ratio relative to the original dense matrix as

$$\mathrm{CR} = 1 - \frac{P_{\mathrm{MORS}}}{mn}. \tag{6}$$

Unlike COMPOT, which reformulates orthogonal sparse dictionary learning in a whitened activation space and maps the learned dictionary back afterwards, MORS defines the factorization directly in the original weight coordinates. Substituting $\widehat{\mathbf{W}} = \mathbf{DS}$ into the loss-aware objective of equation 4 yields

$$\min_{\mathbf{D},\mathbf{S}} \frac{1}{2}\,\mathrm{tr}\left[(\mathbf{DS}-\mathbf{W})^\top \mathbf{A}_\lambda (\mathbf{DS}-\mathbf{W})\mathbf{B}_\lambda\right] \quad \text{s.t.} \quad \mathbf{D}^\top \mathbf{A}_\lambda \mathbf{D} = \mathbf{I}_k, \qquad \|\mathbf{s}_j\|_0 \le s\ \forall j. \tag{7}$$

To expose the geometry of this objective, we use the Cholesky decompositions of the damped positive-definite curvature factors, $\mathbf{A}_\lambda = \mathbf{L}_x\mathbf{L}_x^\top$, $\mathbf{B}_\lambda = \mathbf{L}_g\mathbf{L}_g^\top$, where $\mathbf{L}_x$ and $\mathbf{L}_g$ are lower-triangular Cholesky factors. Using the trace–Frobenius identity, the loss term in equation 7 can equivalently be written as

$$\frac{1}{2}\left\|\mathbf{L}_x^\top(\mathbf{DS}-\mathbf{W})\mathbf{L}_g\right\|_F^2. \tag{8}$$

Thus, the KFAC objective measures reconstruction error under a two-sided curvature transformation, while the optimization variables $\mathbf{D}$ and $\mathbf{S}$ remain in the original weight coordinates.

For the practical MORS solver, including dictionary initialization and both alternating updates, we use the diagonal output-side surrogate

$$\bar{\mathbf{B}}_\lambda = \mathrm{diag}(\mathbf{B}_\lambda) = \mathrm{diag}(\beta_1,\dots,\beta_n), \qquad \bar{\mathbf{B}}_\lambda = \bar{\mathbf{L}}_g\bar{\mathbf{L}}_g^\top, \tag{9}$$

where $\bar{\mathbf{L}}_g = \mathrm{diag}(\sqrt{\beta_1},\dots,\sqrt{\beta_n})$. The full output-side factor is retained for model-wide allocation in Sec. 3.3.

The resulting problem is jointly non-convex. To attack it, we first initialize the dictionary and then solve it by alternating minimization, where each iteration consists of two steps: *(a) Sparse coding*: fix $\mathbf{D}$ and solve for $\mathbf{S}$; and *(b) Dictionary update*: fix $\mathbf{S}$ and solve for $\mathbf{D}$.

Dictionary initialization. We form the curvature-weighted weight matrix

$$\mathbf{M}_{\mathrm{init}} = \mathbf{L}_x^\top \mathbf{W}\bar{\mathbf{L}}_g = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top. \tag{10}$$

The dictionary is initialized from its leading k left singular vectors, mapped back to the original weight coordinates,

$$\mathbf{D}^{(0)} = \mathbf{L}_x^{-\top}\mathbf{U}_{:,1:k}. \tag{11}$$

This initialization satisfies $\mathbf{D}^{(0)\top}\mathbf{A}_\lambda\mathbf{D}^{(0)} = \mathbf{I}_k$ by construction and provides a loss-aware starting point for the subsequent alternating optimization.

Loss-aware sparse coding. For a fixed metric-orthogonal dictionary $\mathbf{D}$, define

$$\mathbf{Z} = \mathbf{D}^\top\mathbf{A}_\lambda\mathbf{W}. \tag{12}$$

Using $\mathbf{D}^\top\mathbf{A}_\lambda\mathbf{D} = \mathbf{I}_k$, the coefficient-dependent part of equation 7 can be completed to

$$\min_{\mathbf{S}\in\mathcal{S}} \frac{1}{2}\operatorname{tr}\left[(\mathbf{S}-\mathbf{Z})^\top(\mathbf{S}-\mathbf{Z})\mathbf{B}_\lambda\right], \tag{13}$$

where $\mathcal{S}$ denotes the chosen sparsity constraint. Without sparsity, the optimum is simply $\mathbf{S} = \mathbf{Z}$. Sparse coding can therefore be interpreted as removing coefficients from the dense optimum while minimizing the resulting increase in the local loss surrogate.

For a general $\mathbf{B}_\lambda$, its off-diagonal entries couple coefficient errors across output columns, preventing independent closed-form support selection. This motivates the diagonal surrogate $\bar{\mathbf{B}}_\lambda$ used by the practical solver above. Under this surrogate, equation 13 reduces to

$$\frac{1}{2}\sum_{j=1}^{n}\beta_j\left\|\mathbf{s}_j-\mathbf{z}_j\right\|_2^2. \tag{14}$$

Under the default per-column constraint $\|\mathbf{s}_j\|_0 \le s$, the exact solution is ordinary hard thresholding,

$$\mathbf{s}_j \leftarrow \mathcal{H}_s(\mathbf{z}_j), \tag{15}$$

where $\mathcal{H}_s$ retains the s largest-magnitude entries. Although the objective is weighted by β_j, this factor is constant within output column j and therefore does not affect its support ordering.

The same formulation also permits a single sparsity budget over the complete projection. In that case, setting coefficient z_{ij} to zero incurs the independent predicted loss increase

$$\Delta_{ij} = \frac{1}{2}\beta_j z_{ij}^2, \tag{16}$$

so a budget of K non-zero coefficients is optimally allocated by retaining the K largest values of Δ_{ij} over the complete coefficient matrix. We compare per-column and projection-level sparsity experimentally in Appendix C.3.

Metric-orthogonal dictionary update. The dictionary subproblem admits a closed-form Procrustes solution for a general output-side metric. In the practical alternating solver, however, we use the same diagonal surrogate $\overline{\mathbf{B}}_\lambda$ introduced for sparse coding so that both block updates optimize a common surrogate objective.

For fixed $\mathbf{S}$, define

$$\mathbf{R} = \mathbf{L}_x^\top\mathbf{W}\overline{\mathbf{B}}_\lambda\mathbf{S}^\top = \mathbf{P}\mathbf{\Sigma}\mathbf{Q}^\top. \tag{17}$$

Using the metric-orthogonality constraint, the dictionary-dependent quadratic term is constant, reducing the update to a generalized orthogonal Procrustes problem. Its global solution is

$$\mathbf{D} \leftarrow \mathbf{L}_x^{-\top}\mathbf{P}\mathbf{Q}^\top, \tag{18}$$

which preserves $\mathbf{D}^\top\mathbf{A}_\lambda\mathbf{D} = \mathbf{I}_k$ by construction.

Starting from the initialized dictionary, MORS alternates between sparse-coding and dictionary-update steps for a fixed number of iterations. Detailed derivations of both subproblems and the convergence properties of the alternating procedure are provided in Appendix C.

3.3 Loss-Aware Dynamic Compression Allocation

A uniform compression ratio ignores substantial differences in redundancy and loss sensitivity across transformer projections. MORS therefore constructs a discrete loss–storage curve for each projection and distributes the global parameter budget according to the predicted loss increase.

Loss-aware candidate scoring. For allocation, we evaluate candidate factorizations under the same second-order KFAC loss surrogate introduced in equation 4, retaining the full output-side curvature. Using the Cholesky factors of the input- and output-side curvature matrices, for each projection i we form

$$\mathbf{M}_i = \mathbf{L}_{x,i}^\top \mathbf{W}_i \mathbf{L}_{g,i} = \mathbf{U}_i \mathbf{\Sigma}_i \mathbf{V}_i^\top, \tag{19}$$

requiring only one SVD per projection.

For a candidate compression ratio c, the storage constraint and fixed k/s ratio determine the dictionary width $k_i(c)$ and sparsity $s_i(c)$. Following a one-step sparse approximation, we retain the leading $k_i(c)$ singular directions and define

$$\mathbf{Z}_{i,c} = \left(\mathbf{\Sigma}_i \mathbf{V}_i^\top\right)_{1:k_i(c),:}, \qquad \widetilde{\mathbf{Z}}_{i,c} = \mathcal{H}_{s_i(c)}\left(\mathbf{Z}_{i,c}\right), \tag{20}$$

where $\mathcal{H}_s$ retains the s largest-magnitude coefficients in each output column. The corresponding predicted loss is

$$\widetilde{q}_i(c) = \frac{1}{2}\left\|\mathbf{M}_i - \mathbf{U}_{i,:,1:k_i(c)}\widetilde{\mathbf{Z}}_{i,c}\right\|_F^2. \tag{21}$$

Since the retained singular vectors are orthonormal, this reduces to

$$\widetilde{q}_i(c) = \frac{1}{2}\left(\|\mathbf{M}_i\|_F^2 - \|\widetilde{\mathbf{Z}}_{i,c}\|_F^2\right). \tag{22}$$

Thus, the complete candidate loss–storage curve can be evaluated from a single SVD, without running the iterative MORS solver for every candidate. The proxy is used only for model-wide allocation; after local compression ratios are selected, each projection is compressed using the full MORS factorization of Section 3.2.

Global budget allocation. Together with its storage cost $P_i(c)$, each candidate defines a point $(P_i(c), \widetilde{q}_i(c))$ on the projection-specific loss–storage curve. Starting from the cheapest candidate for every projection, we iteratively allocate additional parameters according to the marginal predicted loss reduction

$$\eta_i(c \to c') = \frac{\widetilde{q}_i(c) - \widetilde{q}_i(c')}{P_i(c') - P_i(c)}. \tag{23}$$

At each step, the feasible upgrade with the largest η_i is selected until the global storage budget is exhausted. Complete candidate construction and allocation details are provided in Appendix D.

4 Experiments

We evaluate MORS along four complementary dimensions: allocation quality, comparison with state-of-the-art SVD-based methods, compatibility with quantization and end-to-end inference efficiency, and sensitivity to calibration and sparsity choices.

4.1 Experimental Setup

Models. For comparisons with prior dictionary-based compression methods, we evaluate decoder-only language models across several architectures and scales: Llama3.2-1B (Meta AI, 2024), Llama3-8B (Grattafiori et al., 2024), Qwen3-0.6B and Qwen3-8B (Yang et al., 2025). Comparisons with recent SVD-based methods additionally include Llama-7B (Touvron et al., 2023) and Mistral-7B (Jiang et al., 2023). Following previous factorization-based compression methods, we compress the linear projections in self-attention and MLP blocks while leaving token embeddings and the language-model head unchanged. To evaluate whether MORS extends beyond text-only LLMs, we additionally consider the Parakeet-CTC-0.6B for automatic speech recognition (Rekesh et al., 2023) and Qwen3-VL-8B-Instruct for vision-language tasks (Bai et al., 2025).

Compression configuration. Following COMPOT (Makhov et al., 2026b), MORS uses $k/s = 2$, complete or undercomplete dictionaries, and 20 alternating updates. We use † to denote uniform allocation, where the same compression ratio is applied independently to every projection; unmarked methods use their corresponding dynamic allocation strategy. Compression ratio and storage are defined in equation 6 and equation 5.

Table 1: MORS component decomposition across model scales. We report average accuracy (↑), WikiText perplexity (↓), and LAMBADA perplexity (↓). † denotes uniform allocation; otherwise, COMPOT uses its original dynamic allocator and MORS uses the proposed loss-aware allocator. COMPOT results are reproduced under the same evaluation setup. Best compressed results within each model and compression ratio are shown in **bold**.

Method	**Llama3-8B** (dense: 70.36 / 7.26 / 3.09)									**Qwen3-8B** (dense: 70.45 / 12.22 / 4.58)								
	CR = 0.2			CR = 0.3			CR = 0.4			CR = 0.2			CR = 0.3			CR = 0.4		
	Avg.	Wiki.	LAMB.	Avg.	Wiki.	LAMB.	Avg.	Wiki.	LAMB.	Avg.	Wiki.	LAMB.	Avg.	Wiki.	LAMB.	Avg.	Wiki.	LAMB.
COMPOT†	64.52	12.70	5.20	57.95	20.90	8.48	46.74	62.06	41.67	66.40	15.92	5.66	63.04	18.44	6.25	54.73	25.32	9.82
MORS†	64.63	11.16	4.07	57.34	16.93	6.35	46.60	38.96	24.11	69.12	**13.67**	4.46	62.44	18.44	5.64	53.28	24.74	10.40
COMPOT	65.28	12.15	4.61	60.91	17.42	6.56	51.67	42.97	21.68	67.17	15.30	5.52	65.40	17.59	5.30	59.00	22.79	**6.42**
MORS	**67.92**	**9.51**	**3.25**	**64.46**	**12.22**	**3.79**	**56.90**	**19.79**	**7.13**	**69.75**	14.42	**4.17**	**67.79**	**17.08**	**4.52**	**62.02**	**21.54**	8.78
Method	**Llama3.2-1B** (dense: 57.61 / 11.57 / 5.73)									**Qwen3-0.6B** (dense: 50.27 / 26.14 / 24.76)								
	CR = 0.2			CR = 0.3			CR = 0.4			CR = 0.2			CR = 0.3			CR = 0.4		
	Avg.	Wiki.	LAMB.	Avg.	Wiki.	LAMB.	Avg.	Wiki.	LAMB.	Avg.	Wiki.	LAMB.	Avg.	Wiki.	LAMB.	Avg.	Wiki.	LAMB.
COMPOT†	48.51	24.67	20.62	40.83	55.30	84.25	31.59	284.45	2154.54	39.97	1391.85	423.69	35.70	3098.15	3551.39	30.14	6718.34	119228.28
MORS†	48.24	20.69	16.39	42.32	36.77	42.68	35.22	99.79	325.93	44.09	61.07	101.67	37.28	1608.50	397.65	32.65	12735.48	5653.97
COMPOT	50.06	20.71	13.97	45.01	35.68	32.35	37.62	129.28	195.41	40.92	70.35	192.69	39.13	245.34	382.29	34.65	3824.49	4986.30
MORS	**52.52**	**15.85**	**9.42**	**48.16**	**22.67**	**15.01**	**42.35**	**44.01**	**44.83**	**46.40**	**41.22**	**24.44**	**44.87**	**57.56**	**39.41**	**41.51**	**94.59**	**69.38**

Calibration. For fair side-by-side comparisons, we match the calibration regime of the corresponding prior work. COMPOT-style experiments (Makhov et al., 2026b) use 256 FineWeb (Penedo et al., 2024) sequences of length 1024; LEMS-style experiments (Thoma et al., 2026) use 256 WikiText-2 (Merity et al., 2016) sequences of length 2048; and comparisons with recent SVD-based methods, including LACE-SVD and Swift-SVD, use 256 WikiText-2 sequences of length 1024. In all settings, MORS uses a single offline forward–backward pass to accumulate activation and output-gradient statistics, without parameter updates or post-compression finetuning.

Evaluation. We follow the evaluation protocol associated with each comparison. COMPOT-style experiments (Makhov et al., 2026b) report zero-shot accuracy on PIQA (Bisk et al., 2020), HellaSwag (Zellers et al., 2019), LAMBADA (Paperno et al., 2016), ARC-Easy/Challenge (Clark et al., 2018), SciQ (Welbl et al., 2017), RACE (Lai et al., 2017), and MMLU (Hendrycks et al., 2021a), together with WikiText-2 and LAMBADA perplexity. For comparisons with LEMS (Thoma et al., 2026) and recent SVD-based methods, we follow their evaluation setting using downstream zero-shot accuracy together with perplexity on WikiText-2, C4 (Raffel et al., 2020), and Penn Treebank (Marcus et al., 1993). Results obtained under different protocols are compared only within their corresponding tables.

Baselines. Under the COMPOT-style protocol, we compare against SVD-LLM (Wang et al., 2025b), COMPOT (Makhov et al., 2026b), and a controlled KFAC-SVD baseline. KFAC-SVD is the rank-constrained counterpart of the MORS objective and uses the same KFAC statistics and damping while retaining the full output-side metric $\mathbf{B}_\lambda$. Implementation details and additional variants are provided in Appendix B.

For the SVD-based comparisons, we include the baselines reported by LEMS (Thoma et al., 2026), including Uniform, SVD-LLM V2 (Wang et al., 2025a), ARS (Gao et al., 2024), together with recent SVD-based methods such as Swift-SVD (Qi et al., 2026), ZS-SVD (Abbasi et al., 2026), and LACE-SVD (Liu et al., 2026). When following their published calibration and evaluation protocols, we use the corresponding published baseline results directly.

4.2 Main Results

Method decomposition. Table 1 separates the effects of the local factorization and dynamic allocation by comparing COMPOT and MORS under uniform allocation (†) and with their respective allocation strategies. Under uniform allocation, MORS generally improves reconstruction quality, particularly perplexity, while dynamic allocation provides substantial additional gains. Across all four models and compression ratios, full MORS consistently improves average accuracy over full COMPOT and generally yields lower perplexity. Detailed metrics are provided in Appendix G, with additional evaluation on BBH (Suzgun et al., 2023), GPQA (Rein et al., 2024), IFEval (Zhou et al., 2023), MATH-Hard (Hendrycks et al., 2021b), MMLU-Pro (Wang et al., 2024), and MuSR (Sprague et al., 2024) in Appendix G.2.

Table 2: MORS comparison with SVD-based dynamic methods reported by LEMS (Thoma et al., 2026). Best metrics are highlighted with **bold**.

CR	Allocation	Mistral-7B				Llama3-8B				Qwen3-8B			
		Wiki↓	C4↓	PTB↓	Avg. Acc.↑	Wiki↓	C4↓	PTB↓	Avg. Acc.↑	Wiki↓	C4↓	PTB↓	Avg. Acc.↑
Original model		5.25	8.10	27.72	63.95	6.14	9.47	9.90	63.34	9.71	15.52	15.43	62.03
0.20	Uniform	7.14	19.43	64.84	52.35	11.44	52.00	80.06	47.69	12.52	31.85	47.25	53.56
	SVD-LLM V2	7.13	19.35	64.84	52.68	11.40	52.00	80.38	47.70	12.52	31.72	47.44	53.36
	ARS	7.26	11.74	47.71	54.58	11.81	20.44	28.83	54.31	11.58	22.41	25.05	55.66
	LEMS	5.98	11.24	37.09	57.67	8.16	17.52	22.80	55.99	**10.38**	**18.32**	**18.32**	59.28
	MORS†	5.94	10.75	40.57	57.84	8.12	18.87	20.72	56.78	11.09	21.72	23.85	56.98
	MORS	**5.67**	**9.49**	**32.60**	**60.15**	**7.22**	**14.93**	**14.02**	**59.81**	10.75	19.47	19.17	**59.84**
0.40	Uniform	14.38	72.05	208.48	39.10	48.56	475.36	1312.50	34.39	21.68	101.61	195.85	39.63
	SVD-LLM V2	14.90	73.18	209.30	39.29	47.53	467.99	1242.65	34.43	21.42	96.95	189.82	39.74
	ARS	19.43	37.09	166.87	40.83	28.77	56.55	112.03	40.57	29.51	83.91	161.10	40.72
	LEMS	10.58	34.37	124.00	45.60	17.86	81.01	201.28	43.09	15.70	57.22	120.66	45.51
	MORS†	9.20	31.11	102.00	45.00	16.39	95.82	139.42	40.88	15.16	54.60	87.25	44.55
	MORS	**8.24**	**23.12**	**79.44**	**48.42**	**11.09**	**43.19**	**65.86**	**48.78**	**14.47**	**38.72**	**65.86**	**52.30**

Table 3: Post-training quantization on LLaMA-7B at a matched 2.8 GB weight-memory budget. QCR and FCR denote quantization and factorization compression ratios.

Method	QCR	FCR	Wiki. PPL↓
GPTQ-3bit	0.81	–	16.28
SVD-LLM V2 + GPTQ-4bit	0.75	0.25	9.97
COMPOT† + GPTQ-4bit	0.75	0.25	9.93
COMPOT + GPTQ-4bit	0.75	0.25	9.62
MORS† + GPTQ-4bit	0.75	0.25	9.34
MORS + GPTQ-4bit	0.75	0.25	**8.72**

Table 4: A100 decode latency averaged over prompt lengths 128, 512, and 2048. Speedup is relative to SVD-LLM.

Model	CR	Latency (ms/token) ↓		Speedup
		SVD-LLM	MORS	
Llama3-8B	0.3	27.14	22.66	1.20×
	0.4	27.06	23.23	1.17×
Qwen3-8B	0.3	37.99	33.28	1.14×
	0.4	37.61	33.49	1.12×

Comparison with SVD-based methods. Table 2 compares MORS with SVD-based compression and allocation baselines under the evaluation protocol of LEMS (Thoma et al., 2026). MORS achieves the highest average accuracy across all three models at both compression ratios. The advantage becomes more pronounced at $\mathrm{CR} = 0.4$, where MORS improves over LEMS by 2.82, 5.69, and 6.79 percentage points on Mistral-7B, Llama3-8B, and Qwen3-8B, respectively, while also reducing all reported perplexities. At $\mathrm{CR} = 0.2$, MORS also improves average accuracy across all three models; on Qwen3-8B, LEMS retains slightly lower perplexity, showing that improvements in perplexity do not necessarily translate into higher downstream accuracy.

Additional comparisons with recent SVD-based methods, including Swift-SVD (Qi et al., 2026), ZS-SVD (Abbasi et al., 2026), and LACE-SVD (Liu et al., 2026), are provided in Appendix G.4.

Compatibility with Post-Training Quantization. Sparse factorization and quantization remove complementary forms of redundancy. We apply GPTQ (Frantar et al., 2023) after factorization and compare methods at the same reported weight-memory budget on Llama-7B (Touvron et al., 2023). This experiment tests whether the gain from loss-aware sparse factorization is preserved after quantizing the stored factors.

At the matched 2.8 GB weight-memory budget, MORS+GPTQ achieves a WikiText perplexity of 8.72, compared with 9.62 for COMPOT+GPTQ and 9.97 for SVD-LLM V2+GPTQ. Uniform MORS also improves over uniform COMPOT (9.34 vs. 9.93), while dynamic allocation further reduces MORS perplexity from 9.34 to 8.72. Thus, the advantage of MORS is preserved after 4-bit quantization of the stored factors.

Inference efficiency. Previous runtime evidence for sparse dictionary compression focused on a selective deployment regime. In particular, CoSpaDi (Makhov et al., 2026a) compressed only the wide MLP up and gate projections, where the dense-times-sparse operator is especially favorable. Here, we evaluate a more demanding whole-model setting in which MORS compression is applied throughout the transformer rather than only to the up/gate branches.

For practical autoregressive inference on NVIDIA A100 GPUs, we combine MORS with a hardware-aware runtime consisting of A100-specialized MACKO (Macko & Boža, 2026) kernels and system-level MLP optimizations. We compare this runtime with the standard SVD-LLM factorized implementation, where the two low-rank factors are evaluated using dense matrix multiplica-

Table 5: Allocation strategies for MORS on Qwen3-0.6B at 20% compression.

Allocation	Avg. Acc.↑	Wiki. PPL↓	LAMB. PPL↓
Uniform	37.9	565.8	338.9
LEMS	41.9	**37.2**	77.2
LASER	42.8	41.9	60.0
COMPOT	40.9	46.5	99.6
MORS	**44.9**	39.6	**39.5**

Table 6: Calibration data influence on MORS with Qwen3-0.6B at 20% compression.

Data	N	Avg. Acc.↑	Wiki. PPL↓	LAMB. PPL↓
FineWeb	64	43.87	54.45	53.63
FineWeb	128	45.31	46.23	36.27
FineWeb	256	46.40	41.22	**24.44**
WikiText	256	44.11	**37.14**	32.76
Alpaca	256	**47.31**	48.74	34.88

tions. Table 4 reports decode latency averaged across prompt lengths of 128, 512, and 2048 tokens. Despite compressing the full model, MORS is consistently faster than SVD-LLM across all tested settings.

Extension to Other Modalities. We additionally test whether the same factorization extends beyond text-only LLMs. For vision–language evaluation, we compress the language module of Qwen3-VL-8B-Instruct (Bai et al., 2025) and evaluate on MMMU (Yue et al., 2024), OCRBench (Liu et al., 2024), RealWorldQA (xAI, 2024), and MMStar (Chen et al., 2024). At $\mathrm{CR} = 0.3$, dynamic MORS achieves the highest four-benchmark average of 0.6262, compared with 0.6038 for dynamic COMPOT. For speech recognition, we compress NVIDIA Parakeet-CTC-0.6B based on FastConformer (Rekesh et al., 2023). It outperforms existing methods for all compression rates and at higher $\mathrm{CR} = 0.4$, MORS reduces WER from $2.50/5.10$ for COMPOT to $2.24/4.50$ on LibriSpeech test-clean/test-other (Panayotov et al., 2015). Full modality-specific settings and results are reported in Appendix G.6.

4.3 Ablation Studies

Unless stated otherwise, ablations use the COMPOT-style protocol and Qwen3-0.6B. We study calibration sensitivity and model-wide allocation.

Calibration data. Table 6 studies the sensitivity of the allocator to the calibration set on Qwen3-0.6B at $\mathrm{CR} = 0.2$. Increasing the number of FineWeb samples from 64 to 256 consistently improves both downstream accuracy and perplexity. At a fixed budget of 256 sequences, performance remains moderately dependent on the calibration distribution: Alpaca gives the highest average accuracy, while WikiText yields the lowest WikiText perplexity. Overall, the allocator benefits from larger calibration sets without requiring a narrowly matched calibration distribution.

Dynamic allocation. To isolate the effect of model-wide budget allocation, we fix the local MORS factorization and vary only the allocation strategy, comparing uniform allocation with LEMS, LASER, COMPOT, and the proposed MORS allocator. All methods use the same global storage budget and LEMS-style calibration protocol.

As shown in Table 5, MORS achieves the highest average downstream accuracy and lowest LAMBADA. LEMS yields the lowest WikiText-2 perplexity, indicating that perplexity and downstream accuracy need not favor the same allocation. Detailed task-level results are provided in Appendix G.5.

5 Conclusions and Limitations

We introduced **MORS**, a loss-aware sparse dictionary framework for post-training compression. MORS combines metric-orthogonal sparse factorization with a Kronecker-factored empirical Fisher approximation and uses the same loss-aware geometry for model-wide allocation. Across multiple model families and compression ratios, MORS consistently improves over reconstruction-based sparse compression and strong SVD-based baselines, while remaining compatible with quantization and efficient sparse inference.

MORS relies on several approximations. Each projection is evaluated independently around the pretrained model, ignoring cross-projection interactions and activation drift after compression. The curvature is approximated using empirical Fisher/KFAC statistics, and the practical sparse solver further diagonalizes the output-side metric. Finally, the quadratic loss model is local and can become less accurate under severe compression. Addressing these interactions and updating loss statistics during compression are promising directions for future work.

AI USE STATEMENT

Generative AI tools were used to assist with text polishing, grammar and spelling checks, and improving the clarity of the manuscript. They were also used for retrieval and discovery of potentially relevant related work. All AI-assisted edits were reviewed by the authors, and references identified with AI assistance were independently verified against their original sources. The authors take full responsibility for the final content of the paper.

ETHICS STATEMENT

This work studies post-training compression of existing neural networks and does not involve human subjects or collection of personal or sensitive data. Experiments use publicly available pretrained models and benchmark datasets. As with other model-compression techniques, MORS may make deployment of large models less resource-intensive; compressed models can therefore inherit and potentially facilitate the same beneficial or harmful uses, biases, and failure modes as the original models. We do not introduce new training data or modify the intended safety behavior of the evaluated models.

REPRODUCIBILITY STATEMENT

We provide the complete MORS formulation and optimization procedure in Sec. 3, with derivations and dynamic-allocation details in the appendices. Experimental protocols and baselines are described in Sec. 4, default implementation hyperparameters are reported in Appendix F, and complete task-level results and ablations are provided in the corresponding experimental appendices. All reported compression ratios use the storage accounting defined in the main text, and no post-compression finetuning is used.

REFERENCES

Ali Abbasi, Chayne Thrash, Haoran Qin, Shansita Sharma, Sepehr Seifi, and Soheil Kolouri. Zero sum SVD: Balancing loss sensitivity for low rank LLM compression. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=tAdsDOkRBw.

Michal Aharon, Michael Elad, and Alfred M. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54 (11):4311–4322, 2006. doi: 10.1109/TSP.2006.881199.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Chenglong Bao, Jian-Feng Cai, and Hui Ji. Fast sparsity-based orthogonal dictionary learning for image restoration. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3384–3391, 2013.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020. doi: 10.1609/aaai.v34i05.6239.

Viktoriia A. Chekalina, Daniil Moskovskiy, Tatyana Matveeva, Andrey Kuznetsov, and Evgeny Frolov. Scalable kronecker-factored fisher approximation for neural network parameter sensitivity. In *Forty-third International Conference on Machine Learning*, 2026.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large vision-language models? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=evP9mxNNxJ.

Patrick H. Chen, Hsiang-Fu Yu, Inderjit S. Dhillon, and Cho-Jui Hsieh. DRONE: Data-aware low-rank compression for large nlp models. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29321–29334. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/f56de5ef149cf0aedcc8f4797031e229-Abstract.html.

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 preprint arXiv:1803.05457*, 2018.

Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. Open llm leaderboard v2. https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard, 2024.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *International Conference on Learning Representations*, 2023.

Shangqian Gao, Ting Hua, Yen-Chang Hsu, Yilin Shen, and Hongxia Jin. Adaptive rank selections for low-rank approximation of language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 227–241. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-long.13. URL https://aclanthology.org/2024.naacl-long.13/.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=7Bywt2mQsCe.

Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. Language model compression with weighted low-rank factorization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=uPv9Y3gmAI5.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 785–794, 2017. doi: 10.18653/v1/D17-1082.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12):220102, 2024.

Zhuowen Liu, Longkun Hao, Shiyu Feng, Xiaowen Chang, Ruiqun Li, and Changqun Li. LACE-SVD: Loss-aware svd with cumulative error correction for llm compression. *arXiv preprint arXiv:2607.03057*, 2026.

Vladimír Macko and Vladimír Boža. MACKO: Sparse matrix-vector multiplication for low sparsity. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=ah9xkFXCV6.

Denis Makhov, Dmitriy Shopkhoev, Magauiya Zhussip, Ammar Ali, and Stamatios Lefkimmiatis. CoSpaDi: Compressing LLMs via calibration-guided sparse dictionary learning, 2026a.

Denis Makhov, Dmitriy Shopkhoev, Magauiya Zhussip, Ammar Ali, Baher Mohammad, and Stamatios Lefkimmiatis. COMPOT: Calibration-optimized matrix procrustes orthogonalization for transformers compression. *arXiv preprint arXiv:2602.15200*, 2026b. URL https://arxiv.org/abs/2602.15200.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2408–2417. PMLR, 2015. URL https://proceedings.mlr.press/v37/martens15.html.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Meta AI. Llama 3.2: Revolutionizing Edge AI and Vision with Open, Customizable Models. Meta AI, September 2024. URL https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391. Association for Computational Linguistics, 2018. doi: 10.18653/v1/D18-1260. URL https://aclanthology.org/D18-1260/.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5206–5210. IEEE, 2015.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 1525–1534, 2016. doi: 10.18653/v1/P16-1144.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. *arXiv preprint arXiv:2406.17557*, 2024.

Ruoling Qi, Yirui Liu, Xuaner Wu, Xiangyu Wang, Ming Li, Chen Chen, Jian Chen, Yin Chen, and Qizhen Weng. Swift-SVD: Theoretical optimality meets practical efficiency in low-rank LLM compression. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=nAQ4h8FpdM.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Dima Rekesh, Nithin Rao Koluguri, Samuel Kriman, Somshubra Majumdar, Vahid Noroozi, He Huang, Oleksii Hrinchuk, Krishna Puvvada, Ankur Kumar, Jagadeesh Balam, et al. Fast conformer with linearly scalable attention for efficient speech recognition. In *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 1–8. IEEE, 2023.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020. doi: 10.1609/aaai.v34i05.6399.

Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. MuSR: Testing the limits of chain-of-thought with multistep soft reasoning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=jenyYQzue1.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-acl.824. URL https://aclanthology.org/2023.findings-acl.824/.

Moritz Thoma, Maximilian Groezinger, Maximilian Forstenhäusler, Emad Aghajanzadeh, Manoj Rohit Vemparala, Christos Anagnostopoulos, Pierpaolo Mori, Nael Fasfous, Alexander Frickenstein, Daniel Mueller-Gritschneder, and Ulf Schlichtmann. Advancing SVD-based LLM compression via layer-wise error model search. In *Forty-third International Conference on Machine Learning*, Seoul, South Korea, 2026. URL https://openreview.net/forum?id=IjIgNPFuCt.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Haiyu Wang, Yutong Wang, Leshu Li, Yihui Ren, and Sai Qian Zhang. LASER: Loss-aware singular-value decomposition and rank allocation for efficient low-precision vision-language models. *arXiv preprint arXiv:2606.00573*, 2026. URL https://arxiv.org/abs/2606.00573.

Xin Wang, Samiul Alam, Zhongwei Wan, Hui Shen, and Mi Zhang. SVD-LLM v2: Optimizing singular value truncation for large language model compression. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4287–4296. Association for Computational Linguistics, 2025a. doi: 10.18653/v1/2025.naacl-long.217. URL https://aclanthology.org/2025.naacl-long.217/.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. SVD-LLM: Truncation-aware singular value decomposition for large language model compression. In *International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=LNYIUouhdt.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3018. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ad236edc564f3e3156e1b2feafb99a24-Abstract.html. Datasets and Benchmarks Track.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 94–106, 2017. doi: 10.18653/v1/W17-4413.

xAI. RealWorldQA. Hugging Face dataset, 2024. URL https://huggingface.co/datasets/xai-org/RealworldQA.

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 preprint arXiv: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 preprint arXiv:2312.05821*, 2023. URL https://arxiv.org/abs/2312.05821.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9556–9567, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019. doi: 10.18653/v1/P19-1472.

Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pp. 169–186. Springer, 2024.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023. URL https://arxiv.org/abs/2311.07911.

A Loss Approximation and KFAC Derivation

This section provides the derivation of the loss-aware compression objective used by MORS. Consider a linear projection

$$\mathbf{Y} = \mathbf{X}\mathbf{W}, \qquad \mathbf{W} \in \mathbb{R}^{m\times n}, \tag{24}$$

and let

$$\mathbf{E} = \widehat{\mathbf{W}} - \mathbf{W} \tag{25}$$

denote the perturbation introduced by compression. A second-order Taylor expansion of the calibration loss around the pretrained weights gives

$$\Delta\mathcal{L} \approx \langle \mathbf{G}_W, \mathbf{E}\rangle_F + \frac{1}{2}\,\mathrm{vec}(\mathbf{E})^\top \mathbf{H}_{\mathbf{W}}\,\mathrm{vec}(\mathbf{E}), \qquad \mathbf{G}_W = \nabla_{\mathbf{W}}\mathcal{L}. \tag{26}$$

A.1 Empirical Fisher Approximation

Let $\mathbf{g} = \partial\mathcal{L}/\partial\mathbf{y} \in \mathbb{R}^n$ be the gradient with respect to the output of the projection for one calibration token with input $\mathbf{x} \in \mathbb{R}^m$. The corresponding weight gradient is

$$\nabla_{\mathbf{W}}\mathcal{L} = \mathbf{x}\mathbf{g}^\top, \qquad \mathrm{vec}(\nabla_{\mathbf{W}}\mathcal{L}) = \mathbf{g}\otimes\mathbf{x}. \tag{27}$$

The empirical Fisher block is therefore

$$\mathbf{F}_{\mathbf{W}} = \mathbb{E}\left[(\mathbf{g}\otimes\mathbf{x})(\mathbf{g}\otimes\mathbf{x})^\top\right] \tag{28}$$

$$= \mathbb{E}\left[(\mathbf{g}\mathbf{g}^\top)\otimes(\mathbf{x}\mathbf{x}^\top)\right]. \tag{29}$$

Following KFAC (Martens & Grosse, 2015), we approximate the expectation of the Kronecker product by the product of expectations,

$$\mathbf{F}_{\mathbf{W}} \approx \mathbf{B}\otimes\mathbf{A}, \qquad \mathbf{A} = \mathbb{E}[\mathbf{x}\mathbf{x}^\top], \qquad \mathbf{B} = \mathbb{E}[\mathbf{g}\mathbf{g}^\top]. \tag{30}$$

Since the empirical second-moment factors can be singular or poorly conditioned, we use the damped matrices

$$\mathbf{A}_\lambda = \mathbf{A} + \lambda_x\mathbf{I}, \qquad \mathbf{B}_\lambda = \mathbf{B} + \lambda_g\mathbf{I}. \tag{31}$$

Using

$$\mathrm{vec}(\mathbf{E})^\top(\mathbf{B}_\lambda\otimes\mathbf{A}_\lambda)\,\mathrm{vec}(\mathbf{E}) = \mathrm{tr}\left(\mathbf{E}^\top\mathbf{A}_\lambda\mathbf{E}\mathbf{B}_\lambda\right), \tag{32}$$

the Taylor approximation becomes

$$\Delta\mathcal{L} \approx \langle \mathbf{G}_W, \mathbf{E}\rangle_F + \frac{1}{2}\,\mathrm{tr}\left(\mathbf{E}^\top\mathbf{A}_\lambda\mathbf{E}\mathbf{B}_\lambda\right). \tag{33}$$

The first-order term preserves the quadratic form of the objective. Defining

$$\mathbf{W}_{\text{shift}} = \mathbf{W} - \mathbf{A}_\lambda^{-1}\mathbf{G}_W\mathbf{B}_\lambda^{-1}, \tag{34}$$

and completing the square gives

$$\begin{aligned}\Delta\mathcal{L} \approx \frac{1}{2}\operatorname{tr}\left[(\widehat{\mathbf{W}} - \mathbf{W}_{\text{shift}})^\top \mathbf{A}_\lambda (\widehat{\mathbf{W}} - \mathbf{W}_{\text{shift}})\mathbf{B}_\lambda\right] \\ - \frac{1}{2}\operatorname{tr}\left(\mathbf{G}_W^\top \mathbf{A}_\lambda^{-1}\mathbf{G}_W\mathbf{B}_\lambda^{-1}\right).\end{aligned} \tag{35}$$

The second term is independent of $\widehat{\mathbf{W}}$. Thus, retaining the first-order contribution amounts to replacing the original target $\mathbf{W}$ by the shifted target $\mathbf{W}_{\text{shift}}$.

The damping in equation 31 can also be interpreted directly in the original weight coordinates. Expanding the quadratic term,

$$\begin{aligned}&\frac{1}{2}\operatorname{tr}\left[\mathbf{E}^\top(\mathbf{A} + \lambda_x\mathbf{I})\mathbf{E}(\mathbf{B} + \lambda_g\mathbf{I})\right] \\ &= \frac{1}{2}\operatorname{tr}\left(\mathbf{E}^\top\mathbf{A}\mathbf{E}\mathbf{B}\right) + \frac{\lambda_g}{2}\operatorname{tr}\left(\mathbf{E}^\top\mathbf{A}\mathbf{E}\right) \\ &\quad + \frac{\lambda_x}{2}\operatorname{tr}\left(\mathbf{E}^\top\mathbf{E}\mathbf{B}\right) + \frac{\lambda_x\lambda_g}{2}\|\mathbf{E}\|_F^2.\end{aligned} \tag{36}$$

Thus, damping the two KFAC factors introduces both one-sided quadratic regularization terms and an isotropic Frobenius penalty. In the activation-only case $\mathbf{B}_\lambda = \mathbf{I}$, this reduces to

$$\frac{1}{2}\operatorname{tr}\left[\mathbf{E}^\top(\mathbf{A} + \lambda_x\mathbf{I})\mathbf{E}\right] = \frac{1}{2}\|\mathbf{A}^{\frac{1}{2}}\mathbf{E}\|_F^2 + \frac{\lambda_x}{2}\|\mathbf{E}\|_F^2, \tag{37}$$

which corresponds exactly to Frobenius regularization of the weight perturbation.

In the experiments, we omit the first-order contribution and use the second-order surrogate

$$Q(\mathbf{E}) = \frac{1}{2}\operatorname{tr}\left(\mathbf{E}^\top\mathbf{A}_\lambda\mathbf{E}\mathbf{B}_\lambda\right). \tag{38}$$

This corresponds to treating the pretrained checkpoint as locally close to a stationary point and avoids shifting the compression target according to a finite-sample calibration gradient. Setting $\mathbf{B}_\lambda = \mathbf{I}$ removes output-side curvature weighting and recovers the activation-aware objective governed solely by $\mathbf{A}_\lambda$.

B KFAC-SVD Baseline

To isolate the effect of the sparse dictionary representation from the loss-aware curvature model itself, we construct a controlled low-rank counterpart of MORS using the same KFAC statistics and damping. Unlike the sparse MORS problem, the rank-constrained counterpart admits a closed-form solution through a two-sided curvature transformation followed by truncated SVD.

B.1 Two-sided loss-aware low-rank approximation

Recall the final MORS loss surrogate from equation 38,

$$\mathcal{Q}(\widehat{\mathbf{W}} - \mathbf{W}) = \frac{1}{2}\operatorname{tr}\left[(\widehat{\mathbf{W}} - \mathbf{W})^\top \mathbf{A}_\lambda (\widehat{\mathbf{W}} - \mathbf{W})\mathbf{B}_\lambda\right]. \tag{39}$$

KFAC-SVD replaces the sparse dictionary constraint of MORS by a rank constraint,

$$\min_{\widehat{\mathbf{W}}} \frac{1}{2}\operatorname{tr}\left[(\widehat{\mathbf{W}} - \mathbf{W})^\top \mathbf{A}_\lambda (\widehat{\mathbf{W}} - \mathbf{W})\mathbf{B}_\lambda\right] \quad \text{s.t.} \quad \operatorname{rank}(\widehat{\mathbf{W}}) \le r. \tag{40}$$

Using the Cholesky decompositions

$$\mathbf{A}_\lambda = \mathbf{L}_x\mathbf{L}_x^\top, \qquad \mathbf{B}_\lambda = \mathbf{L}_g\mathbf{L}_g^\top, \tag{41}$$

Equation 40 is equivalent to

$$\min_{\operatorname{rank}(\widehat{\mathbf{W}})\le r} \frac{1}{2}\left\|\mathbf{L}_x^\top(\widehat{\mathbf{W}}-\mathbf{W})\mathbf{L}_g\right\|_F^2. \tag{42}$$

Define

$$\widetilde{\mathbf{W}}=\mathbf{L}_x^\top\mathbf{W}\mathbf{L}_g, \qquad \widetilde{\widehat{\mathbf{W}}}=\mathbf{L}_x^\top\widehat{\mathbf{W}}\mathbf{L}_g. \tag{43}$$

Because both Cholesky factors are invertible, this transformation preserves rank and reduces the problem to

$$\min_{\operatorname{rank}(\widetilde{\widehat{\mathbf{W}}})\le r} \frac{1}{2}\left\|\widetilde{\widehat{\mathbf{W}}}-\widetilde{\mathbf{W}}\right\|_F^2. \tag{44}$$

Let

$$\widetilde{\mathbf{W}}=\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top. \tag{45}$$

By the Eckart–Young theorem, the optimal rank-r approximation is

$$\widetilde{\mathbf{W}}_r=\mathbf{U}_r\mathbf{\Sigma}_r\mathbf{V}_r^\top, \tag{46}$$

and mapping it back to the original coordinates gives

$$\boxed{\widehat{\mathbf{W}}_{\text{KFAC-SVD}}=\mathbf{L}_x^{-\top}\mathbf{U}_r\mathbf{\Sigma}_r\mathbf{V}_r^\top\mathbf{L}_g^{-1}}. \tag{47}$$

For inference, we store the approximation as two rank-r factors,

$$\mathbf{F}_L=\mathbf{L}_x^{-\top}\mathbf{U}_r\mathbf{\Sigma}_r^{1/2}, \qquad \mathbf{F}_R=\mathbf{\Sigma}_r^{1/2}\mathbf{V}_r^\top\mathbf{L}_g^{-1}, \tag{48}$$

so that $\widehat{\mathbf{W}}_{\text{KFAC-SVD}}=\mathbf{F}_L\mathbf{F}_R$. The stored parameter count is $r(m+n)$ for an $m\times n$ projection. For a target compression ratio CR, we select the largest integer rank satisfying

$$r(m+n)\le(1-\text{CR})mn. \tag{49}$$

B.2 Relation to MORS

KFAC-SVD is derived from the same general KFAC loss geometry as MORS but imposes a different representation constraint. KFAC-SVD restricts the compressed matrix to a rank-r subspace shared across all output columns, whereas MORS uses a metric-orthogonal dictionary with sparse coefficients. It therefore provides a controlled low-rank reference for separating the effect of the sparse representation from that of curvature-aware weighting.

A further distinction is that KFAC-SVD can retain the full output-side factor $\mathbf{B}_\lambda$ because the two-sided transformation preserves the standard truncated-SVD solution. The practical MORS alternating solver uses the diagonal output-curvature surrogate introduced in Sec. 3.2. Unless stated otherwise, KFAC-SVD uses the same calibration examples, empirical-Fisher statistics, and damping as the corresponding MORS experiment.

C Metric-Orthogonal Sparse Dictionary Learning

MORS represents a projection as

$$\widehat{\mathbf{W}}=\mathbf{D}\mathbf{S}, \qquad \mathbf{D}\in\mathbb{R}^{m\times k}, \qquad \mathbf{S}\in\mathbb{R}^{k\times n}, \tag{50}$$

and starts from the full loss-aware objective

$$\min_{\mathbf{D},\mathbf{S}} \frac{1}{2}\operatorname{tr}\left[(\mathbf{D}\mathbf{S}-\mathbf{W})^\top\mathbf{A}_\lambda(\mathbf{D}\mathbf{S}-\mathbf{W})\mathbf{B}_\lambda\right] \quad \text{s.t.} \quad \mathbf{D}^\top\mathbf{A}_\lambda\mathbf{D}=\mathbf{I}_k, \qquad \mathbf{S}\in\mathcal{S}. \tag{51}$$

The optimization variables remain in the original weight coordinates. We use the Cholesky decompositions of the damped curvature factors

$$\mathbf{A}_\lambda=\mathbf{L}_x\mathbf{L}_x^\top, \qquad \mathbf{B}_\lambda=\mathbf{L}_g\mathbf{L}_g^\top, \tag{52}$$

where $\mathbf{L}_x$ and $\mathbf{L}_g$ are lower-triangular. The transformed quantities below are used only to construct analytical block updates and the initialization.

C.1 Dictionary Initialization

The general objective in equation 51 is defined using the full output-side curvature $\mathbf{B}_\lambda$. The practical MORS solver uses the diagonal surrogate

$$\bar{\mathbf{B}}_\lambda = \mathrm{diag}(\mathbf{B}_\lambda) = \bar{\mathbf{L}}_g \bar{\mathbf{L}}_g^\top, \qquad \bar{\mathbf{L}}_g = \mathrm{diag}(\sqrt{\beta_1}, \dots, \sqrt{\beta_n}). \tag{53}$$

We use the same surrogate for dictionary initialization and the subsequent alternating updates.

We form

$$\mathbf{M}_{\mathrm{init}} = \mathbf{L}_x^\top \mathbf{W} \bar{\mathbf{L}}_g = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top. \tag{54}$$

The initial dictionary is

$$\mathbf{D}^{(0)} = \mathbf{L}_x^{-\top} \mathbf{U}_{:,1:k}, \tag{55}$$

which satisfies $(\mathbf{D}^{(0)})^\top \mathbf{A}_\lambda \mathbf{D}^{(0)} = \mathbf{I}_k$. Starting from $\mathbf{D}^{(0)}$, MORS alternates sparse-coding and dictionary-update steps.

C.2 Alternating updates

C.2.1 Loss-aware sparse coding

For a fixed metric-orthogonal dictionary $\mathbf{D}$, define

$$\mathbf{Z} = \mathbf{D}^\top \mathbf{A}_\lambda \mathbf{W} \in \mathbb{R}^{k \times n}. \tag{56}$$

Using $\mathbf{D}^\top \mathbf{A}_\lambda \mathbf{D} = \mathbf{I}_k$, the coefficient-dependent part of equation 51 can be completed to

$$\min_{\mathbf{S} \in \mathcal{S}} \frac{1}{2} \mathrm{tr}\left[(\mathbf{S} - \mathbf{Z})^\top (\mathbf{S} - \mathbf{Z}) \mathbf{B}_\lambda \right]. \tag{57}$$

Without sparsity, the optimum is $\mathbf{S} = \mathbf{Z}$. Using the practical diagonal output-side metric introduced above, the sparse-coding objective then becomes

$$\mathcal{J}(\mathbf{S} \mid \mathbf{D}) = \frac{1}{2} \sum_{j=1}^{n} \beta_j \|\mathbf{s}_j - \mathbf{z}_j\|_2^2 + \mathrm{const}. \tag{58}$$

Under the default per-column constraint

$$\mathcal{S}_{\mathrm{col}} = \{\mathbf{S} : \|\mathbf{s}_j\|_0 \le s,\ \forall j\}, \tag{59}$$

β_j is constant within each output column, so the exact update is ordinary hard thresholding,

$$\boxed{\mathbf{s}_j^\star = \mathcal{H}_s(\mathbf{z}_j)}. \tag{60}$$

C.2.2 Metric-orthogonal dictionary update

For consistency with sparse coding, the practical alternating solver uses the same diagonal surrogate $\overline{\mathbf{B}}_\lambda$ in the dictionary block. For fixed $\mathbf{S}$, expanding the resulting objective gives

$$\begin{aligned} \mathcal{J}(\mathbf{D}, \mathbf{S}) = \frac{1}{2} \mathrm{tr}\left(\mathbf{S}^\top \mathbf{D}^\top \mathbf{A}_\lambda \mathbf{D} \mathbf{S} \overline{\mathbf{B}}_\lambda \right) \\ - \mathrm{tr}\left(\mathbf{D}^\top \mathbf{A}_\lambda \mathbf{W} \overline{\mathbf{B}}_\lambda \mathbf{S}^\top \right) + \mathrm{const}. \end{aligned} \tag{61}$$

The first term is constant under metric orthogonality. Therefore, with $\widetilde{\mathbf{D}} = \mathbf{L}_x^\top \mathbf{D}$, the update reduces to an orthogonal Procrustes problem. Define

$$\mathbf{R} = \mathbf{L}_x^\top \mathbf{W} \overline{\mathbf{B}}_\lambda \mathbf{S}^\top = \mathbf{P}\mathbf{\Sigma}_R \mathbf{Q}^\top. \tag{62}$$

The global block minimizer is

$$\boxed{\mathbf{D}^\star = \mathbf{L}_x^{-\top} \mathbf{P} \mathbf{Q}^\top}, \tag{63}$$

which preserves $\mathbf{D}^\top \mathbf{A}_\lambda \mathbf{D} = \mathbf{I}_k$. The dictionary subproblem also admits a Procrustes solution with the full $\mathbf{B}_\lambda$; using $\overline{\mathbf{B}}_\lambda$ here simply ensures that both practical block updates optimize the same surrogate.

With the initialization in equation 55, one iteration is therefore

$$\mathbf{S}^{(t+1)} \leftarrow \mathrm{SparseCode}(\mathbf{D}^{(t)}), \qquad \mathbf{D}^{(t+1)} \leftarrow \mathrm{DictUpdate}(\mathbf{S}^{(t+1)}). \tag{64}$$

We repeat these two steps for a fixed number of iterations. In the implementation, one final sparse-coding update is applied to the returned dictionary after the last dictionary update.

Table 7: Coefficient-sparsity granularity on Qwen3-0.6B. Per-column and per-projection variants use the same total coefficient budget. Best result within each allocation setting is shown in **bold**.

CR	Projection alloc.	Coeff. sparsity	Avg. Acc.↑	Wiki. PPL↓	LAMB. PPL↓
0.2	Uniform	Per-column	**44.09**	**61.07**	**101.67**
		Per-projection	41.56	94.28	143.25
	Dynamic	Per-column	**46.40**	**41.22**	**24.44**
		Per-projection	45.91	46.09	29.41
0.3	Uniform	Per-column	**37.28**	**1608.50**	**397.65**
		Per-projection	36.37	2738.38	566.00
	Dynamic	Per-column	**44.87**	**57.56**	**39.41**
		Per-projection	43.57	66.06	50.91
0.4	Uniform	Per-column	**32.65**	12735.48	**5653.97**
		Per-projection	31.28	**12047.33**	13093.66
	Dynamic	Per-column	**41.51**	**94.59**	**69.38**
		Per-projection	40.76	112.77	113.50

C.3 Sparsity granularity

The per-column constraint used above fixes the same number of non-zero coefficients in every output column. The diagonal surrogate also permits a single coefficient budget over the complete projection. Starting from the dense optimum $\mathbf{Z}$, setting coefficient z_{ij} to zero increases the surrogate by

$$\delta_{ij} = \frac{\beta_j}{2} z_{ij}^2. \tag{65}$$

For a projection-level budget

$$\mathcal{S}_{\text{proj}} = \{\mathbf{S} : \|\mathbf{S}\|_0 \le K\}, \qquad K = sn, \tag{66}$$

the exact minimizer retains the K coefficients with the largest δ_{ij} over the complete coefficient matrix. Thus, projection-level sparsity is more flexible under the local diagonal surrogate, whereas per-column sparsity enforces a balanced number of atoms for every output column.

Table 7 compares the two choices on Qwen3-0.6B using the same dictionary width, total coefficient budget, $k/s = 2$, and 20 alternating updates. Per-column sparsity gives higher average accuracy in every tested setting and generally lower perplexity, despite the projection-level rule being the exact minimizer of the diagonal sparse-coding proxy under a global coefficient budget. We therefore use per-column sparsity throughout the main experiments.

C.4 Convergence of alternating MORS

Once the diagonal surrogate is introduced, the practical alternating solver minimizes

$$\mathcal{J}(\mathbf{D}, \mathbf{S}) = \frac{1}{2} \operatorname{tr}\left[(\mathbf{DS} - \mathbf{W})^\top \mathbf{A}_\lambda (\mathbf{DS} - \mathbf{W}) \overline{\mathbf{B}}_\lambda\right], \tag{67}$$

with $\mathbf{A}_\lambda \succ 0$ and $\overline{\mathbf{B}}_\lambda \succ 0$.

Proposition 1 (Monotonic descent). Let $\{\mathbf{D}^{(t)}, \mathbf{S}^{(t)}\}_{t \ge 0}$ be generated by alternating the exact sparse-coding update under either $\mathcal{S}_{\text{col}}$ or $\mathcal{S}_{\text{proj}}$ and the Procrustes update in equation 63. Then

$$\mathcal{J}(\mathbf{D}^{(t+1)}, \mathbf{S}^{(t+1)}) \le \mathcal{J}(\mathbf{D}^{(t)}, \mathbf{S}^{(t)}) \tag{68}$$

for every iteration t.

Proof. For fixed $\mathbf{D}^{(t)}$, the sparse-coding rule globally minimizes equation 58 under the selected sparsity set, hence

$$\mathcal{J}(\mathbf{D}^{(t)}, \mathbf{S}^{(t+1)}) \le \mathcal{J}(\mathbf{D}^{(t)}, \mathbf{S}^{(t)}). \tag{69}$$

For fixed $\mathbf{S}^{(t+1)}$, equation 63 globally solves the metric-Procrustes subproblem, so

$$\mathcal{J}(\mathbf{D}^{(t+1)}, \mathbf{S}^{(t+1)}) \le \mathcal{J}(\mathbf{D}^{(t)}, \mathbf{S}^{(t+1)}). \tag{70}$$

Combining the two inequalities proves the result. □

Corollary 1 (Convergence of objective values). The sequence $\{\mathcal{J}(\mathbf{D}^{(t)}, \mathbf{S}^{(t)})\}_{t\geq 0}$ converges.

Proof. By Proposition 1 the sequence is monotonically non-increasing. Since equation 67 is non-negative, it is bounded below and therefore converges. □

Proposition 2 (Block-coordinate optimality of fixed points). Any fixed point $(\mathbf{D}^\star, \mathbf{S}^\star)$ of the alternating updates is block-coordinate optimal with respect to the selected sparsity set:

$$\mathbf{S}^\star \in \arg\min_{\mathbf{S}\in\mathcal{S}} \mathcal{J}(\mathbf{D}^\star, \mathbf{S}), \tag{71}$$

$$\mathbf{D}^\star \in \arg\min_{\mathbf{D}^\top \mathbf{A}_\lambda \mathbf{D}=\mathbf{I}_k} \mathcal{J}(\mathbf{D}, \mathbf{S}^\star). \tag{72}$$

At a fixed point, each exact block update leaves its corresponding variable unchanged, so each block is globally optimal with the other fixed. This does not imply global optimality of the jointly non-convex problem or convergence of the iterates to a unique point; hard-thresholding ties and non-unique Procrustes solutions can produce equivalent block minimizers.

D Dynamic Allocation Details

MORS separates local factor optimization from model-wide budget allocation. For every compressible projection, the allocator evaluates a discrete set of candidate compression ratios using a fast one-step sparse proxy in the full two-sided KFAC geometry. The resulting loss–storage curves are then combined under a single model-wide storage budget.

D.1 Candidate loss–storage curves

For projection $\mathbf{W}_i \in \mathbb{R}^{m_i\times n_i}$, let the damped KFAC factors admit Cholesky decompositions

$$\mathbf{A}_{\lambda,i} = \mathbf{L}_{x,i}\mathbf{L}_{x,i}^\top, \qquad \mathbf{B}_{\lambda,i} = \mathbf{L}_{g,i}\mathbf{L}_{g,i}^\top. \tag{73}$$

The corresponding local quadratic loss surrogate is

$$\mathcal{Q}_i(\mathbf{E}_i) = \frac{1}{2}\left\|\mathbf{L}_{x,i}^\top \mathbf{E}_i \mathbf{L}_{g,i}\right\|_F^2. \tag{74}$$

For allocation, we retain the full output-side factor and form

$$\mathbf{M}_i = \mathbf{L}_{x,i}^\top \mathbf{W}_i \mathbf{L}_{g,i} = \mathbf{U}_i \mathbf{\Sigma}_i \mathbf{V}_i^\top. \tag{75}$$

Only one SVD of $\mathbf{M}_i$ is required for all candidate compression ratios of projection i.

Let $\mathcal{G}$ denote the candidate grid. For $c \in \mathcal{G}$, the target compression ratio together with the fixed k/s ratio determines the dictionary width $k_i(c)$ and coefficient sparsity $s_i(c)$ using the same storage accounting as the final compressor. A compressed candidate has

$$P_i(c) = m_i k_i(c) + s_i(c) n_i + \frac{k_i(c) n_i}{16}, \tag{76}$$

while the uncompressed candidate has $P_i(0) = m_i n_i$.

One-step sparse candidate. For a candidate with $k = k_i(c)$, retain the leading k singular directions and define

$$\mathbf{Z}_{i,c} = \left(\mathbf{\Sigma}_i \mathbf{V}_i^\top\right)_{1:k,:}, \qquad \widetilde{\mathbf{Z}}_{i,c} = \mathcal{H}_{s_i(c)}(\mathbf{Z}_{i,c}), \tag{77}$$

where $\mathcal{H}_s$ keeps the s largest-magnitude coefficients in every output column. This gives the transformed candidate approximation

$$\widehat{\mathbf{M}}_{i,c} = (\mathbf{U}_i)_{:,1:k}\widetilde{\mathbf{Z}}_{i,c}. \tag{78}$$

We score it directly under the two-sided KFAC loss geometry,

$$q_i(c) = \frac{1}{2}\left\|\mathbf{M}_i - (\mathbf{U}_i)_{:,1:k}\widetilde{\mathbf{Z}}_{i,c}\right\|_F^2. \tag{79}$$

Because $(\mathbf{U}_i)_{:,1:k}$ is orthonormal and $\widetilde{\mathbf{Z}}_{i,c}$ is obtained by masking entries of $\mathbf{Z}_{i,c} = (\mathbf{U}_i)_{:,1:k}^\top \mathbf{M}_i$, this reduces to

$$\boxed{q_i(c) = \frac{1}{2}\left(\|\mathbf{M}_i\|_F^2 - \|\widetilde{\mathbf{Z}}_{i,c}\|_F^2\right)}. \tag{80}$$

Thus all candidate scores are obtained from one curvature-weighted SVD per projection, without running the iterative MORS solver for every candidate. The thresholded transformed coefficients are used only to score candidate budgets and are not returned as the deployed coefficient matrix. Once local compression ratios are selected, every projection is compressed with the full alternating MORS solver from Appendix C.

Equivalently, if $\sigma_{i,r}$ is the r-th singular value and $v_{i,jr}$ the corresponding element of $\mathbf{V}_i$, then

$$q_i(c) = \frac{1}{2}\left[\|\mathbf{M}_i\|_F^2 - \sum_{j=1}^{n_i} \mathrm{TopSum}_{s_i(c)}\left(\left\{\sigma_{i,r}^2 v_{i,jr}^2\right\}_{r=1}^{k_i(c)}\right)\right], \tag{81}$$

where TopSum_s returns the sum of the s largest entries.

Relation to the COMPOT allocator. The COMPOT allocator (Makhov et al., 2026b) uses a spectral proxy derived from the original weight matrix and then applies a discrete model-wide budget-allocation procedure. MORS retains the candidate grid, storage accounting, and one-SVD-per-projection structure, but replaces the original spectral score by the sparse candidate loss in equation 79. The resulting proxy accounts jointly for the two-sided KFAC geometry, the candidate dictionary width, and coefficient sparsity.

Together with its storage cost, each valid candidate defines a point

$$\Gamma_i = \left\{\left(P_i(c), q_i(c), c\right) : c \in \mathcal{G}\right\} \tag{82}$$

on the projection-specific loss–storage curve. Candidates resolving to the same storage cost are deduplicated by keeping the one with the smallest predicted loss, and the remaining points are sorted from cheapest to most expensive.

D.2 Global budget allocation

For target model-wide compression ratio ρ, the available storage is

$$P_{\text{budget}} = (1-\rho)\sum_i m_i n_i. \tag{83}$$

We initialize every projection at the cheapest point on its loss–storage curve. If c is the current candidate for projection i and c' is the next more expensive candidate, the upgrade requires

$$\Delta P_i = P_i(c') - P_i(c) \tag{84}$$

and provides predicted loss reduction

$$\Delta q_i = q_i(c) - q_i(c'). \tag{85}$$

We rank feasible upgrades by marginal utility

$$\eta_i(c \to c') = \frac{q_i(c) - q_i(c')}{P_i(c') - P_i(c)}. \tag{86}$$

At each step, the feasible upgrade with the largest η_i is selected until no further candidate fits within the remaining budget. This is a greedy approximation to the discrete global allocation problem.

E Inference Complexity and Storage Cost

Consider a projection $\mathbf{W} \in \mathbb{R}^{m\times n}$ compressed as $\widehat{\mathbf{W}} = \mathbf{DS}$, where $\mathbf{D} \in \mathbb{R}^{m\times k}$ and each column of $\mathbf{S} \in \mathbb{R}^{k\times n}$ contains s non-zero coefficients. For one input vector $\mathbf{x}$, MORS evaluates

$$\mathbf{x}\widehat{\mathbf{W}} = (\mathbf{xD})\mathbf{S}. \tag{87}$$

Algorithm 1 MORS loss-aware dynamic allocation

Require: Projection weights $\{\mathbf{W}_i\}$, KFAC factors $\{\mathbf{A}_{\lambda,i}, \mathbf{B}_{\lambda,i}\}$, target global compression ρ, candidate grid $\mathcal{G}$, fixed ratio $\gamma = k/s$
Ensure: Local compression ratios $\{c_i^\star\}$

for each compressible projection i **do**
  $\mathbf{L}_{x,i}\mathbf{L}_{x,i}^\top \leftarrow \mathbf{A}_{\lambda,i}$
  $\mathbf{L}_{g,i}\mathbf{L}_{g,i}^\top \leftarrow \mathbf{B}_{\lambda,i}$
  $\mathbf{M}_i \leftarrow \mathbf{L}_{x,i}^\top \mathbf{W}_i \mathbf{L}_{g,i}$
  $\mathbf{M}_i \leftarrow \mathbf{U}_i \mathbf{\Sigma}_i \mathbf{V}_i^\top$ ▷ one SVD per projection
  $e_i \leftarrow \|\mathbf{M}_i\|_F^2$, $\Gamma_i \leftarrow \varnothing$
  for each $c \in \mathcal{G}$ **do**
    if $c = 0$ **then**
      add $(c, m_i n_i, 0)$ to Γ_i
      continue
    end if
    $(k, s, p) \leftarrow \textsc{ResolveKS}(c, \gamma, m_i, n_i)$
    if $k < 1$ **or** $k > \min(m_i, n_i)$ **then**
      continue
    end if
    $\mathbf{Z} \leftarrow (\mathbf{\Sigma}_i \mathbf{V}_i^\top)_{1:k,:}$
    $\widetilde{\mathbf{Z}} \leftarrow \mathcal{H}_s(\mathbf{Z})$
    $q \leftarrow \frac{1}{2}\left(e_i - \|\widetilde{\mathbf{Z}}\|_F^2\right)$
    add (c, p, q) to Γ_i
  end for
  deduplicate Γ_i by storage, retaining the minimum-q candidate
  sort Γ_i from cheapest to most expensive
  $a_i \leftarrow 1$ ▷ cheapest operating point
end for
$P_{\text{budget}} \leftarrow (1-\rho)\sum_i m_i n_i$
$P_{\text{used}} \leftarrow \sum_i P_i(a_i)$
while a feasible upgrade exists **do**
  for each projection i with a next point on Γ_i **do**
    $\Delta P_i \leftarrow P_i(a_i + 1) - P_i(a_i)$
    $\eta_i \leftarrow \big(q_i(a_i) - q_i(a_i + 1)\big)/\Delta P_i$
  end for
  $i^\star \leftarrow \arg\max_{i:\, P_{\text{used}} + \Delta P_i \le P_{\text{budget}}} \eta_i$
  if no feasible $i^\star$ exists **then**
    break
  end if
  $P_{\text{used}} \leftarrow P_{\text{used}} + \Delta P_{i^\star}$
  $a_{i^\star} \leftarrow a_{i^\star} + 1$
end while
return $c_i^\star \leftarrow \Gamma_i[a_i].c$ for all i

The dense multiplication by $\mathbf{D}$ requires $\mathcal{O}(mk)$ operations, while multiplication by the sparse $\mathbf{S}$ requires $\mathcal{O}(sn)$. Therefore,

$$C_{\text{MORS}} = \mathcal{O}(mk + sn). \tag{88}$$

Let $\gamma = k/s$ and let ρ denote the target compression ratio. Ignoring support metadata, the retained numerical parameter budget is $(1-\rho)mn$, so

$$mk + sn = k\left(m + \frac{n}{\gamma}\right) = (1-\rho)mn. \tag{89}$$

For an SVD factorization of rank r at the same compression ratio,

$$r(m+n) = (1-\rho)mn. \tag{90}$$

Table 8: Default MORS hyperparameters.

Hyperparameter	Value
Dictionary-to-sparsity ratio k/s	2
Coefficient sparsity	Per-column
Alternating updates	20
Local damping	10^{-4}
Allocator damping	10^{-6}
Candidate CR step	0.01
Candidate CR range	$[0, 0.99]$

Hence MORS and SVD have the same ideal arithmetic complexity at matched compression,

$$C_{\mathrm{MORS}} = C_{\mathrm{SVD}} = \mathcal{O}((1-\rho)mn)\,. \tag{91}$$

Our reported compression ratio additionally accounts for the binary support mask of $\mathbf{S}$. With FP16 dictionary and coefficient values, the FP16-equivalent MORS storage is

$$P_{\mathrm{MORS}} = mk + sn + \frac{kn}{16}. \tag{92}$$

Therefore, at target compression ratio ρ,

$$k = \frac{(1-\rho)mn}{m + n/\gamma + n/16}, \tag{93}$$

and the numerical multiply–accumulate count is

$$mk + sn = (1-\rho)mn\frac{m + n/\gamma}{m + n/\gamma + n/16}. \tag{94}$$

The small difference from $(1-\rho)mn$ arises because part of the reported storage budget is occupied by support metadata rather than numerical values.

These expressions describe theoretical arithmetic complexity rather than wall-clock latency. Sparse execution additionally depends on memory traffic, kernel efficiency, batching, and operator fusion; we therefore evaluate end-to-end inference latency directly on the target hardware.

F Implementation Hyperparameters

Unless stated otherwise, we use the same MORS hyperparameters across all models and compression ratios. The default configuration is summarized in Table 8.

G Full Experimental Results

This appendix provides the complete task-level results underlying the compact comparisons in the main paper. We first report the COMPOT-style evaluation used for the controlled factorization comparison, followed by additional reasoning and instruction-following benchmarks. We then expand the LEMS-style comparison using its complete evaluation suite and conclude with complementary cross-paper comparisons to recent SVD-based methods and results on speech and vision–language models.

G.1 Full COMPOT-Style Evaluation

Tables 9 and 10 provide the complete task-level results under the COMPOT-style evaluation protocol, extending Table 1 with SVD-LLM and the controlled KFAC-SVD baseline from Appendix B. SVD-LLM and KFAC-SVD use a uniform projection-wise compression ratio, whereas † denotes uniform allocation for COMPOT and MORS. Unmarked COMPOT and MORS use their respective dynamic allocation strategies. All reproduced methods use the same calibration and evaluation

Table 9: Full COMPOT-style evaluation on Llama3.2-1B and Qwen3-0.6B. Accuracy metrics are reported in percent.

Method	CR	PIQA	Hella.	LAMB.	ARC-e	ARC-c	SciQ	RACE	MMLU	Avg.↑	Wiki.↓	LAMB. PPL↓
Llama3.2-1B	–	74.53	63.66	62.95	60.47	36.20	88.30	37.79	37.00	57.61	11.57	5.73
SVD-LLM	0.2	62.13	36.43	24.43	36.03	25.09	64.90	29.00	22.98	37.62	169.00	170.00
KFAC-SVD	0.2	63.87	40.11	30.58	38.30	23.12	72.50	30.24	23.02	40.22	44.84	58.92
COMPOT†	0.2	69.70	52.09	42.09	49.87	30.20	81.80	33.88	28.46	48.51	24.67	20.62
MORS†	0.2	69.91	51.66	43.74	50.80	28.58	83.40	33.68	24.17	48.24	20.69	16.39
COMPOT	0.2	70.62	54.57	47.64	52.48	31.14	81.60	35.69	26.70	50.06	20.71	13.97
MORS	0.2	72.47	56.17	53.27	58.50	32.68	87.90	35.41	23.74	**52.52**	**15.85**	**9.42**
SVD-LLM	0.3	55.65	30.06	9.10	30.47	21.50	45.90	25.83	23.23	30.22	590.00	2470.00
KFAC-SVD	0.3	59.09	33.41	16.51	32.20	21.25	63.90	26.03	23.07	34.43	118.13	392.84
COMPOT†	0.3	64.20	41.04	27.50	39.81	26.02	73.20	31.39	23.50	40.83	55.30	84.25
MORS†	0.3	64.69	42.93	33.07	40.53	26.37	77.20	30.33	23.42	42.32	36.77	42.68
COMPOT	0.3	67.03	46.79	37.90	47.14	27.05	77.30	33.11	23.78	45.01	35.68	32.35
MORS	0.3	69.75	49.26	45.31	52.53	28.24	83.20	32.34	24.61	**48.16**	**22.67**	**15.01**
SVD-LLM	0.4	51.80	27.25	1.26	26.85	22.87	32.30	24.40	22.97	26.21	1580.00	33000.00
KFAC-SVD	0.4	56.58	29.81	9.16	29.84	20.99	52.50	25.45	22.97	30.91	309.09	1752.02
COMPOT†	0.4	57.02	31.05	7.86	31.65	22.10	53.00	26.99	23.06	31.59	284.45	2154.54
MORS†	0.4	59.03	34.01	16.77	32.66	22.18	66.20	27.94	22.94	35.22	99.79	325.93
COMPOT	0.4	62.89	37.51	21.31	37.96	25.26	62.40	30.62	22.97	37.62	129.28	195.41
MORS	0.4	64.64	40.89	32.60	44.28	24.66	78.20	30.62	22.94	**42.35**	**44.01**	**44.83**
Qwen3-0.6B	–	67.46	47.37	40.03	55.56	34.30	83.50	33.68	40.22	50.27	26.14	24.76
SVD-LLM	0.2	56.96	32.97	12.61	33.59	24.83	54.90	29.47	26.14	33.94	3130.00	7270.00
KFAC-SVD	0.2	59.14	35.12	20.34	33.08	23.72	60.20	32.92	27.60	36.52	3286.08	980.01
COMPOT†	0.2	62.24	40.62	15.89	41.29	29.61	70.80	33.01	26.29	39.97	1391.85	423.69
MORS†	0.2	63.28	41.77	25.69	45.96	29.27	79.70	34.16	32.90	44.09	61.07	101.67
COMPOT	0.2	61.15	42.58	20.63	42.85	28.84	74.00	32.34	24.93	40.92	70.35	192.69
MORS	0.2	65.83	43.57	43.08	47.69	28.67	78.20	31.29	32.86	**46.40**	**41.22**	**24.44**
SVD-LLM	0.3	52.18	29.00	8.71	28.20	23.81	42.10	29.09	24.43	29.69	19100.00	134000.00
KFAC-SVD	0.3	57.62	31.09	11.68	30.68	21.76	50.60	31.00	23.92	32.29	6989.99	24430.66
COMPOT†	0.3	58.22	34.84	13.62	34.05	27.05	59.90	31.67	26.26	35.70	3098.15	3551.39
MORS†	0.3	59.03	37.41	19.89	37.92	25.60	59.00	30.72	28.64	37.28	1608.50	397.65
COMPOT	0.3	61.26	38.26	20.76	40.49	27.13	65.20	32.73	27.18	39.13	245.34	382.29
MORS	0.3	64.36	40.50	40.15	47.77	29.44	78.00	31.67	27.09	**44.87**	**57.56**	**39.41**
SVD-LLM	0.4	53.32	26.94	3.80	26.52	23.89	32.50	25.36	23.58	26.99	72400.00	5420000.00
KFAC-SVD	0.4	50.76	27.61	4.95	25.63	24.15	36.00	26.89	24.47	27.56	17024.48	1429645.18
COMPOT†	0.4	54.52	29.71	8.36	28.83	23.55	44.30	28.33	23.49	30.14	6718.34	119228.28
MORS†	0.4	54.84	32.31	14.54	30.72	23.21	51.20	29.00	25.40	32.65	12735.48	5653.97
COMPOT	0.4	56.86	32.74	17.48	31.73	25.09	57.40	31.20	24.70	34.65	3824.49	4986.30
MORS	0.4	60.88	36.28	40.15	38.47	25.94	73.20	30.33	26.79	**41.51**	**94.59**	**69.38**

setup. Accuracy metrics are reported in percent. Bold marks the best compressed result within each model and compression ratio for average accuracy and perplexity.

The controlled low-rank baselines show that the improvement is not explained only by incorporating curvature information. KFAC-SVD uses the same loss-aware KFAC geometry as MORS but retains a rank-constrained representation, whereas MORS combines the curvature metric with sparse dictionary coefficients. Across the four models, the sparse formulations provide substantially better quality than both SVD-LLM and KFAC-SVD, particularly as compression increases. Dynamic MORS further improves over its uniformly allocated counterpart in average accuracy at nearly all settings.

G.2 Recent Reasoning and Instruction-Following Benchmarks

We additionally evaluate MORS using the Open LLM Leaderboard v2 suite (Fourrier et al., 2024), comprising BBH (Suzgun et al., 2023), GPQA (Rein et al., 2024), IFEval (Zhou et al., 2023), MATH-Hard (Hendrycks et al., 2021b), MMLU-Pro (Wang et al., 2024), and MuSR (Sprague et al., 2024). MATH-Hard corresponds to the level-5 subset of the MATH benchmark. These tasks complement the conventional zero-shot suite with more challenging reasoning, mathematical problem solving, scientific knowledge, and instruction-following evaluation. All metrics are reported in percent.

Table 10: Full COMPOT-style evaluation on Llama3-8B and Qwen3-8B. Accuracy metrics are reported in percent.

Method	CR	PIQA	Hella.	LAMB.	ARC-e	ARC-c	SciQ	RACE	MMLU	Avg.↑	Wiki.↓	LAMB. PPL↓
Llama3-8B	–	80.69	79.13	75.57	77.69	53.50	93.90	40.29	62.15	70.36	7.26	3.09
SVD-LLM	0.2	71.06	58.37	59.25	55.51	33.96	86.40	35.50	32.62	54.08	40.70	10.90
KFAC-SVD	0.2	72.69	57.60	53.23	55.81	29.69	87.20	37.99	33.86	53.51	24.59	11.01
COMPOT†	0.2	77.75	72.68	69.36	71.00	44.45	89.80	40.96	50.14	64.52	12.70	5.20
MORS†	0.2	78.94	71.78	71.61	70.24	42.92	91.40	40.48	49.67	64.63	11.16	4.07
COMPOT	0.2	78.02	73.51	70.50	71.34	45.31	91.00	40.10	52.42	65.28	12.15	4.61
MORS	0.2	79.82	74.47	75.80	76.09	47.70	92.10	41.05	56.35	**67.92**	**9.51**	**3.25**
SVD-LLM	0.3	65.78	46.40	38.06	41.88	27.65	70.00	31.77	27.15	43.58	147.00	60.90
KFAC-SVD	0.3	66.54	45.35	29.52	41.92	25.85	78.70	32.82	23.10	42.98	54.67	65.01
COMPOT†	0.3	74.37	64.64	60.92	61.03	38.05	87.10	38.56	38.91	57.95	20.90	8.48
MORS†	0.3	74.32	63.68	63.48	59.93	33.79	88.50	38.95	36.03	57.34	16.93	6.35
COMPOT	0.3	76.55	67.23	64.31	69.36	41.55	90.10	38.37	39.82	60.91	17.42	6.56
MORS	0.3	77.58	69.77	72.00	73.32	43.77	91.20	40.10	47.97	**64.46**	**12.22**	**3.79**
SVD-LLM	0.4	60.28	34.50	11.43	32.37	24.49	44.20	25.74	23.05	32.01	549.00	1300.00
KFAC-SVD	0.4	60.12	35.82	10.95	32.66	22.44	59.60	29.19	23.03	34.23	151.67	574.32
COMPOT†	0.4	68.23	51.42	38.75	46.93	30.97	75.40	34.35	27.87	46.74	62.06	41.67
MORS†	0.4	69.10	50.74	41.82	44.57	27.82	79.30	34.74	24.72	46.60	38.96	24.11
COMPOT	0.4	70.95	55.84	44.94	56.61	33.45	85.40	36.56	29.64	51.67	42.97	21.68
MORS	0.4	74.10	60.76	59.01	63.34	34.56	88.50	37.89	37.04	**56.90**	**19.79**	**7.13**
Qwen3-8B	–	77.69	74.94	64.08	80.68	56.74	95.70	40.86	72.95	70.45	12.22	4.58
SVD-LLM	0.2	73.83	63.91	62.18	68.69	45.73	90.10	40.48	54.73	62.46	20.60	6.40
KFAC-SVD	0.2	71.60	60.00	57.11	60.40	38.74	87.90	41.24	51.32	58.54	21.15	7.81
COMPOT†	0.2	75.46	72.19	62.53	70.71	48.98	92.40	42.87	66.04	66.40	15.92	5.66
MORS†	0.2	77.53	72.28	65.77	79.38	54.01	94.90	40.38	68.67	69.12	**13.67**	4.46
COMPOT	0.2	76.93	73.13	62.29	72.81	50.77	94.50	40.10	66.86	67.17	15.30	5.52
MORS	0.2	76.93	71.70	67.67	79.76	56.23	95.10	41.82	68.81	**69.75**	14.42	**4.17**
SVD-LLM	0.3	70.35	55.22	53.77	59.26	37.12	87.20	38.37	44.82	55.76	27.40	10.70
KFAC-SVD	0.3	68.44	51.67	45.14	49.37	31.40	84.40	38.76	39.14	51.04	27.92	17.72
COMPOT†	0.3	75.03	65.93	62.45	66.33	44.20	90.10	42.11	58.20	63.04	18.44	6.25
MORS†	0.3	73.34	62.48	62.97	67.13	44.71	92.70	39.52	56.67	62.44	18.44	5.64
COMPOT	0.3	76.12	68.77	64.02	71.17	47.87	93.10	42.39	59.74	65.40	17.59	5.30
MORS	0.3	75.73	67.33	66.66	79.04	51.37	94.20	44.11	63.85	**67.79**	**17.08**	**4.52**
SVD-LLM	0.4	66.27	44.58	37.90	45.03	28.07	77.30	35.31	29.01	45.44	43.00	35.90
KFAC-SVD	0.4	63.98	43.17	30.02	39.31	25.09	75.70	36.46	25.36	42.39	42.63	70.77
COMPOT†	0.4	70.89	56.40	54.71	53.66	34.90	84.30	38.66	44.35	54.73	25.32	9.82
MORS†	0.4	70.18	53.25	52.88	50.63	32.42	83.00	38.66	45.22	53.28	24.74	10.40
COMPOT	0.4	72.52	60.28	61.83	60.69	41.89	88.30	40.19	46.32	59.00	22.79	**6.42**
MORS	0.4	73.45	60.24	59.21	71.59	44.97	90.40	41.63	54.66	**62.02**	**21.54**	8.78

The effect of dynamic allocation varies across individual downstream tasks. This is expected because the allocator optimizes a local loss proxy estimated on general-domain calibration data rather than task-specific evaluation metrics. Consequently, a lower calibration loss does not necessarily imply improvement on every downstream capability.

G.3 Detailed LEMS-Style Comparison

Tables 12–14 expand the LEMS-style comparison from Table 2. Following LEMS (Thoma et al., 2026), we report WikiText-2, C4, and Penn Treebank perplexity together with OpenBookQA (Mihaylov et al., 2018), ARC-Easy, ARC-Challenge, WinoGrande (Sakaguchi et al., 2020), HellaSwag, and PIQA accuracy. Non-MORS baseline values are taken from the detailed LEMS evaluation. Their retained fractions of 0.8 and 0.6 are written here as compression ratios 0.2 and 0.4, respectively. MORS† uses uniform allocation, while MORS uses the proposed dynamic allocator.

Across all three models, the detailed task-level results follow the aggregate trend in the main paper. At CR $= 0.4$, the MORS allocator produces the largest gains, while at CR $= 0.2$ Qwen3-8B illustrates that perplexity and downstream accuracy need not induce the same ordering: LEMS has slightly lower perplexity, whereas MORS has the higher six-task average accuracy.

Table 11: MORS on recent reasoning and instruction-following benchmarks. † denotes uniform allocation.

Model	Method	CR	BBH	GPQA	IFEval	MATH-Hard	MMLU-Pro	MuSR
Qwen3-0.6B	MORS†	0.2	30.12	26.26	18.24	1.13	12.72	33.47
	MORS		30.08	27.01	0.66	0.15	12.16	33.73
	MORS†	0.3	30.60	25.59	11.07	0.00	11.25	34.92
	MORS		29.51	25.92	14.82	0.83	11.06	34.52
	MORS†	0.4	29.09	24.24	17.26	0.00	11.35	34.79
	MORS		29.06	26.01	17.39	0.45	11.20	34.13
Qwen3-8B	MORS†	0.2	31.31	27.18	36.22	4.46	14.05	40.74
	MORS		29.58	28.69	40.35	8.31	11.90	41.80
	MORS†	0.3	34.58	26.01	24.46	1.66	21.09	40.74
	MORS		33.78	27.01	28.50	2.64	16.76	41.40
	MORS†	0.4	36.63	25.25	17.52	1.13	18.62	36.51
	MORS		44.84	24.83	19.18	1.51	26.70	38.76

Table 12: Detailed LEMS-style evaluation on Mistral-7B. Accuracy metrics are reported in percent.

		Perplexity ↓			Accuracy ↑						
Method	CR	Wiki	C4	PTB	OpenB.	ARC-e	ARC-c	WinoG.	HellaS.	PIQA	Avg.↑
Original	–	5.25	8.10	27.72	35.00	80.00	51.00	75.00	62.00	81.00	63.96
Uniform	0.2	7.14	19.43	64.84	26.00	70.00	36.00	69.00	43.00	70.00	52.35
SVD-LLM V2	0.2	7.13	19.35	64.84	27.00	70.00	37.00	69.00	43.00	70.00	52.68
ARS	0.2	7.26	11.74	47.71	28.00	71.00	38.00	68.00	48.00	75.00	54.58
LEMS	0.2	5.98	11.24	37.09	34.00	76.00	41.00	69.00	50.00	77.00	57.67
MORS†	0.2	5.94	10.75	40.57	29.00	75.84	41.30	71.27	52.32	77.31	57.84
MORS	0.2	**5.67**	**9.49**	**32.60**	29.80	78.91	45.48	70.80	56.52	79.38	**60.15**
Uniform	0.4	14.38	72.05	208.48	19.00	47.00	21.00	57.00	32.00	59.00	39.10
SVD-LLM V2	0.4	14.90	73.18	209.30	19.00	47.00	21.00	58.00	32.00	59.00	39.29
ARS	0.4	19.43	37.09	166.87	19.00	49.00	23.00	57.00	34.00	64.00	40.83
LEMS	0.4	10.58	34.37	124.00	22.00	59.00	28.00	62.00	36.00	66.00	45.60
MORS†	0.4	9.20	31.11	102.00	20.00	60.14	26.37	62.12	35.78	65.61	45.00
MORS	0.4	**8.24**	**23.12**	**79.44**	22.20	67.85	32.08	59.83	39.82	68.72	**48.42**

G.4 Comparison with Contemporary SVD-Based Methods

Recent work has independently explored loss- and sensitivity-aware SVD compression. Table 15 provides a complementary cross-paper comparison with Swift-SVD (Qi et al., 2026), ZS-SVD (Abbasi et al., 2026), and LACE-SVD (Liu et al., 2026) on Llama-7B. Unlike the controlled comparisons above, calibration and evaluation details are not fully matched across these methods. Baseline values follow the corresponding publicly reported results, whereas MORS is calibrated using 256 WikiText sequences of length 1024. Metrics not reported by a baseline are shown as "–".

At moderate compression, MORS is competitive with the recent SVD-based methods across both perplexity and downstream accuracy. At CR $= 0.6$, the relative ordering becomes less consistent, which is compatible with the expected limitations of a projection-wise local loss approximation under severe perturbations. Because the protocols are not fully matched, we use this table only as complementary evidence and rely on the controlled comparisons above for direct conclusions.

G.5 Detailed Dynamic Allocation Results

Table 16 expands the allocation ablation from Table 5 to the complete downstream evaluation suite and all three compression ratios. The local MORS factorization is fixed throughout; only the model-wide allocation rule is varied. All methods use the same global storage budget and 256 WikiText-2 calibration sequences of length 2048.

Table 13: Detailed LEMS-style evaluation on Llama3-8B. Accuracy metrics are reported in percent.

		Perplexity ↓			Accuracy ↑						
Method	CR	Wiki	C4	PTB	OpenB.	ARC-e	ARC-c	WinoG.	HellaS.	PIQA	Avg.↑
Original	–	6.14	9.47	9.90	34.00	81.00	52.00	74.00	61.00	79.00	63.33
Uniform	0.2	11.44	52.00	80.06	24.00	61.00	31.00	64.00	38.00	68.00	47.69
SVD-LLM V2	0.2	11.40	52.00	80.38	24.00	62.00	30.00	64.00	38.00	68.00	47.70
ARS	0.2	11.81	20.44	28.83	28.00	71.00	39.00	68.00	47.00	72.00	54.31
LEMS	0.2	8.16	17.52	22.80	32.00	74.00	39.00	69.00	48.00	75.00	55.99
MORS†	0.2	8.12	18.87	20.72	29.60	74.16	40.44	70.17	50.91	75.35	56.78
MORS	0.2	**7.22**	**14.93**	**14.02**	31.80	77.74	44.88	71.90	54.55	78.02	**59.81**
Uniform	0.4	48.56	475.36	1312.50	16.00	33.00	20.00	54.00	28.00	55.00	34.39
SVD-LLM V2	0.4	47.53	467.99	1242.65	16.00	33.00	19.00	54.00	28.00	56.00	34.43
ARS	0.4	28.77	56.55	112.03	19.00	50.00	22.00	55.00	33.00	64.00	40.57
LEMS	0.4	17.86	81.01	201.28	21.00	51.00	26.00	62.00	35.00	64.00	43.09
MORS†	0.4	16.39	95.82	139.42	19.80	49.66	22.18	57.62	33.71	62.30	40.88
MORS	0.4	**11.09**	**43.19**	**65.86**	25.20	65.95	30.20	61.09	39.91	70.29	**48.78**

Table 14: Detailed LEMS-style evaluation on Qwen3-8B. Accuracy metrics are reported in percent.

		Perplexity ↓			Accuracy ↑						
Method	CR	Wiki	C4	PTB	OpenB.	ARC-e	ARC-c	WinoG.	HellaS.	PIQA	Avg.↑
Original	–	9.71	15.52	15.43	31.00	84.00	56.00	68.00	57.00	77.00	62.04
Uniform	0.2	12.52	31.85	47.25	30.00	71.00	42.00	65.00	44.00	69.00	53.56
SVD-LLM V2	0.2	12.52	31.72	47.44	30.00	70.00	42.00	64.00	44.00	69.00	53.36
ARS	0.2	11.58	22.41	25.05	27.00	77.00	46.00	65.00	45.00	74.00	55.66
LEMS	0.2	**10.38**	**18.32**	**18.32**	30.00	80.00	51.00	68.00	51.00	76.00	59.28
MORS†	0.2	11.09	21.72	23.85	28.80	76.52	47.95	65.90	49.30	73.39	56.98
MORS	0.2	10.75	19.47	19.17	31.00	81.86	52.82	67.40	51.67	74.27	**59.84**
Uniform	0.4	21.68	101.61	195.85	19.00	43.00	24.00	59.00	33.00	61.00	39.63
SVD-LLM V2	0.4	21.42	96.95	189.82	19.00	43.00	23.00	59.00	33.00	61.00	39.74
ARS	0.4	29.51	83.91	161.10	18.00	50.00	23.00	58.00	32.00	63.00	40.72
LEMS	0.4	15.70	57.22	120.66	23.00	60.00	29.00	60.00	35.00	66.00	45.51
MORS†	0.4	15.16	54.60	87.25	22.40	52.53	28.75	62.19	36.77	64.64	44.55
MORS	0.4	**14.47**	**38.72**	**65.86**	28.20	71.13	40.87	64.48	40.97	68.17	**52.30**

G.6 Additional Modalities

We additionally evaluate whether the same factorization transfers beyond decoder-only text models.

G.6.1 Speech Recognition

For ASR task extension, we utilize NVIDIA Parakeet-CTC0.6b model based on the FastConformer (Rekesh et al., 2023). We compress the self-attention and feed-forward linear projections in all 24 FastConformer encoder blocks, leaving the convolutional modules and CTC output head unchanged. We use 256 samples from the LibriSpeech (Panayotov et al., 2015) clean validation split for calibration. Table 17 reports word error rate (WER) on LibriSpeech test-clean and test-other using greedy CTC decoding.

G.6.2 Vision–Language Models

For Qwen3-VL-8B-Instruct (Bai et al., 2025), we compress the language module following the same factorization pipeline and use 256 MathVerse (Zhang et al., 2024) samples for calibration. Table 18 reports MMMU (Yue et al., 2024), OCRBench (Liu et al., 2024), RealWorldQA (xAI, 2024), and MMStar (Chen et al., 2024).

Table 15: Comparison with contemporary SVD-based compression methods on Llama-7B. The uncompressed model is included as reference. Accuracy metrics are reported in percent, and Avg. is the unweighted mean over the six downstream tasks. Protocols are not fully matched across methods, so this table is intended as complementary cross-paper evidence rather than a controlled head-to-head comparison. "–" denotes an unreported metric.

		Perplexity ↓		Accuracy ↑						
Method	CR	WikiText	C4	OpenBook	ARC-e	WinoG.	HellaS.	ARC-c	PIQA	Avg.↑
Original	–	5.67	7.33	34.60	75.38	69.46	56.95	41.98	78.78	59.52
Swift-SVD (dynamic)	0.2	7.84	11.15	27.00	65.00	68.00	48.00	–	73.00	–
ZS-SVD		6.74	10.74	31.00	70.00	68.00	49.00	37.00	74.00	54.83
LACE-SVD		7.39	10.99	26.00	62.00	66.00	45.00	33.00	71.00	50.50
MORS		**6.27**	**8.91**	30.00	71.46	70.17	52.17	36.60	75.68	**56.01**
Swift-SVD (dynamic)	0.4	13.29	21.92	23.00	51.00	62.00	38.00	–	67.00	–
ZS-SVD		11.44	34.13	23.00	52.00	62.00	35.00	26.00	64.00	43.67
LACE-SVD		12.00	21.30	23.00	50.00	60.00	35.00	26.00	62.00	42.67
MORS		**9.34**	**19.17**	25.00	57.41	60.77	38.42	27.05	65.83	**45.75**
Swift-SVD (dynamic)	0.6	62.32	137.77	16.00	30.00	53.00	28.00	–	57.00	–
ZS-SVD		45.17	212.57	15.00	31.00	54.00	28.00	21.00	55.00	34.00
LACE-SVD		**32.57**	**72.15**	16.00	32.00	52.00	29.00	23.00	55.00	**34.50**
MORS		50.50	184.70	13.80	32.32	52.01	27.81	18.52	55.01	33.24

Table 16: Detailed comparison of allocation strategies for MORS on Qwen3-0.6B. All methods use the same local MORS factorization and differ only in model-wide allocation. Accuracy metrics are reported in percent. Best compressed results within each compression ratio are shown in **bold**.

Allocation	CR	Accuracy ↑									Perplexity ↓	
		PIQA	Hella.	LAMB.	ARC-e	ARC-c	SciQ	RACE	MMLU	Avg.	Wiki.	LAMB.
Dense	–	67.46	47.37	40.03	55.56	34.30	83.50	33.68	40.22	50.27	26.14	24.76
Uniform	0.2	58.43	38.99	20.30	38.09	26.54	63.20	31.67	25.97	37.90	565.77	338.89
LEMS		**63.55**	40.66	28.22	**45.79**	29.18	71.10	31.39	25.15	41.88	**37.18**	77.24
LASER		63.00	41.87	31.92	43.81	28.33	73.80	32.63	26.89	42.78	41.93	60.00
COMPOT		60.61	40.52	26.86	44.11	28.58	68.00	30.91	27.96	40.94	46.54	99.58
MORS		63.28	**42.13**	**37.43**	44.19	**29.95**	**78.80**	**33.21**	**30.12**	**44.89**	39.57	**39.51**
Uniform	0.3	55.82	33.89	14.36	32.11	23.29	56.10	28.52	26.17	33.78	1226.25	1392.35
LEMS		59.68	36.90	23.93	37.92	25.68	55.50	28.52	23.35	36.44	**47.75**	187.13
LASER		60.12	38.40	28.04	38.01	**27.65**	63.80	29.67	24.29	38.75	57.27	145.79
COMPOT		58.38	36.63	23.04	36.95	25.17	54.80	30.14	23.94	36.13	116.44	259.63
MORS		**60.88**	**38.45**	**36.00**	**38.89**	26.88	**69.00**	**31.20**	**26.68**	**41.00**	54.31	**69.10**
Uniform	0.4	53.48	30.84	7.55	28.75	23.72	48.00	27.75	**25.09**	30.65	10270.21	23410.78
LEMS		57.24	34.11	19.10	34.18	23.04	47.50	29.00	23.06	33.40	**66.38**	436.76
LASER		58.22	34.44	22.72	35.94	**25.09**	60.90	**30.33**	22.90	36.32	85.29	603.17
COMPOT		56.20	33.12	14.69	31.61	23.04	54.00	29.57	24.60	33.35	2709.40	2352.68
MORS		**58.49**	**35.08**	**29.13**	**38.13**	23.98	**63.00**	28.61	24.43	**37.61**	80.92	**239.24**

Table 17: WER results for Parakeet-CTC-0.6B on LibriSpeech test-clean and test-other at different compression ratios (CR). † denotes uniform allocation.

Methods	CR	WER (LibriSpeech test-clean)	WER (LibriSpeech test-other)
Parakeet 0.6B CTC	–	1.929	3.855
SVD-LLM	0.2	2.374	4.791
KFAC-SVD		2.431	5.038
COMPOT†		2.121	4.239
COMPOT		2.060	4.127
MORS†		2.075	4.239
MORS		**1.932**	**3.926**
SVD-LLM	0.3	2.851	5.777
KFAC-SVD		3.161	6.349
COMPOT†		2.326	4.648
COMPOT		2.176	4.407
MORS		2.320	4.736
MORS†		**2.012**	**4.048**
SVD-LLM	0.4	3.831	7.619
KFAC-SVD		4.690	9.285
COMPOT†		2.720	5.565
COMPOT		2.503	5.103
MORS		2.960	6.026
MORS†		**2.242**	**4.501**

Table 18: Comparison of Qwen3-VL-8B Instruct across multiple compression settings on four multimodal benchmarks: MMMU, OCRBench, RealWorldQA, and MMStar. The original uncompressed model is reported as a reference, while the strongest result among the compressed variants is indicated in **bold**. Note that † means uniform allocation.

Method	CR	MMMU val ↑	OCRBench ↑	RealWorldQA ↑	MMStar ↑	Average ↑
Qwen3-VL-8B Instruct	–	0.5256	0.8259	0.6967	0.6315	0.6699
SVD-LLM	0.2	0.4022	0.6800	0.6601	0.5102	0.5631
KFAC-SVD		0.3878	0.7370	0.6431	0.5021	0.5675
COMPOT†		0.4867	0.7540	0.6601	0.5693	0.6175
COMPOT		0.4900	0.7620	0.6784	0.5739	0.6261
MORS†		0.4644	0.7980	0.6719	0.5663	0.6252
MORS		**0.5044**	**0.8050**	**0.6797**	**0.6071**	**0.6491**
SVD-LLM	0.3	0.3189	0.5940	0.5817	0.4336	0.4821
KFAC-SVD		0.3178	0.6900	0.5739	0.4696	0.5128
COMPOT†		0.4333	0.7120	**0.6693**	0.5422	0.5892
COMPOT		0.4433	0.7440	0.6654	0.5625	0.6038
MORS†		0.3989	**0.7580**	0.6261	0.5407	0.5809
MORS		**0.4967**	**0.7580**	0.6641	**0.5861**	**0.6262**
SVD-LLM	0.4	0.2833	0.4060	0.4562	0.3482	0.3734
KFAC-SVD		0.2778	0.5330	0.4784	0.3521	0.4103
COMPOT†		0.3389	0.6130	0.6026	0.4363	0.4977
COMPOT		0.3389	0.6160	**0.6392**	0.4843	0.5196
MORS†		0.3233	**0.6740**	0.5869	0.4841	0.5171
MORS		**0.4311**	0.6260	0.6170	**0.5408**	**0.5537**