H-SCALE: HESSIAN-GUIDED SCALE REFINEMENT FOR NVFP4 SUB-BYTE LLM INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

The NVIDIA Blackwell architecture, with native support for the ultra-fine-grained NVFP4 format, opens new opportunities for accelerating large language model (LLM) inference. NVFP4's micro-block design, such as a group size of 16, offers strong representational flexibility for capturing local weight distributions and isolating outliers, but it also introduces a large and highly sensitive space of per-group scaling factors. Existing post-training quantization (PTQ) methods primarily focus on refining quantized weight values, leaving this scale-selection step underexplored. To address this gap, we propose **H-Scale**, a lightweight post-processing method for NVFP4 per-group scale refinement. Instead of minimizing plain weight reconstruction error, H-Scale selects hardware-valid group scales using a diagonal second-order proxy derived from calibration activations, thereby targeting layer output perturbation more directly. It is designed as a drop-in replacement for RTN-style scale selection in diverse NVFP4 pipelines, requires only modest offline calibration, and introduces strictly zero overhead at inference time. Under a fixed evaluation protocol, experiments on mainstream LLMs show that H-Scale generally improves a broad range of NVFP4 baselines and brings several variants closer to the BF16 reference.

1 INTRODUCTION

Large language models (LLMs) have demonstrated strong capabilities across diverse natural language tasks (Brown et al., 2020; Touvron et al., 2023). Yet their large parameter counts impose severe demands on memory bandwidth and inference throughput (Pope et al., 2023). Post-training quantization (PTQ) has therefore become an important technique for compressing high-precision model tensors (BF16/FP16) into lower-bit formats that are faster and cheaper to execute (Gholami et al., 2021). The field is also undergoing a hardware-software co-design shift from integer-based formats (INT8/INT4) toward 4-bit floating-point (FP4) representations (Rouhani et al., 2023).

The NVIDIA Blackwell architecture introduces native tensor core acceleration for the **NVFP4** format (NVIDIA, 2025). NVFP4 typically uses an ultra-fine-grained structure: weights are organized into micro-blocks (e.g., $g = 16$), where an E2M1 4-bit value grid is coupled with a shared E4M3 8-bit scale per block and a tensor-level FP32 global scale (Panferov et al., 2026). This fine granularity can better capture local weight distributions and isolate outliers, yet it also creates a large space of per-group scales whose calibration strongly affects final accuracy. In NVFP4, each 16-value group introduces its own scale, so scale miscalibration can become a major source of error even when the FP4 value grid is fixed. This challenge is not fully addressed by existing PTQ methods, and recent studies suggest that techniques which transfer well to earlier quantization settings, such as Hadamard-based preprocessing, become less effective under the NVFP4 microscaling regime (Egiazarian et al., 2025).

Advanced PTQ methods such as GPTQ (Frantar et al., 2023) and SpinQuant (Liu et al., 2025) reduce quantization error by reconstructing or regularizing low-bit weights. However, these methods remain largely *weight-centric*: they primarily compensate error by modifying the quantized weight values themselves. We argue that in the NVFP4 setting, the large space of *scaling factors* is a comparably important and still underexplored calibration target. To this end, we introduce **H-Scale**, a targeted refinement method for per-group scale selection in fine-grained formats. Instead of optimizing the discrete weight representation directly, H-Scale uses a diagonal second-order proxy computed from

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.

calibration activations to choose group scales. By shifting the scale-selection objective from plain weight reconstruction error to output-aware weighted reconstruction, H-Scale reduces the rounding noise induced by the coarse E2M1 grid more effectively.

Designed as a lightweight post-processing algorithm, H-Scale can be seamlessly grafted onto existing NVFP4 baselines. It runs in minutes and, crucially, introduces **zero additional cost** at inference time. Our main contributions are threefold:

1. We identify per-group scale selection as an important bottleneck for NVFP4 fidelity and propose H-Scale, a diagonal-Hessian-weighted scale refinement method for ultra-fine-grained quantization.
2. We formulate a drop-in scale-selection step, i.e., for each group, it enumerates a short window of neighboring FP8 local scales around the baseline choice and keeps the one that minimizes Hessian-weighted reconstruction error. The quantized weights remain in native NVFP4, so inference cost is unchanged.
3. Through comprehensive evaluations on state-of-the-art LLMs, we report that H-Scale generally yields higher average performance over diverse baseline algorithms. On complex reasoning benchmarks, H-Scale reduces the average gap between NVFP4 and full-precision BF16 models in several settings.

2 Related Work

Post-Training Quantization (PTQ) for LLMs. PTQ remains the dominant paradigm for LLM compression because of its computational efficiency. The push toward 4-bit precision exposed the limits of naive rounding and motivated stronger error-compensation methods. GPTQ (Frantar et al., 2023) reconstructs weights iteratively using second-order information, AWQ (Lin et al., 2024) protects salient channels through activation-aware scaling, and GPTAQ (Li et al., 2025) further mitigates errors from outlier activations. Orthogonal transform-based methods such as SpinQuant (Liu et al., 2025) instead learn rotations to smooth weight distributions before quantization. These methods established the viability of sub-8-bit quantization, but they mainly improve low-bit fidelity through weight reconstruction, weight reparameterization, or activation-aware protection. H-Scale instead chooses the per-group scales of micro-block formats with a diagonal second-order proxy, leaving the quantized weight codes to the original rounding rule.

Hardware-Co-Designed and Fine-Grained Quantization. Native 4-bit floating-point arithmetic, as in NVIDIA Blackwell, has catalyzed NVFP4-specific algorithms, where the fine-grained group size ($g = 16$) offers high representational capacity but increases scale sensitivity (Panferov et al., 2026). MR-GPTQ (Egiazarian et al., 2025) adapts GPTQ to the microscaling NVFP4 layout and provides a strong baseline. 4over6 (Cook et al., 2025) regularizes value distributions through adaptive sub-block formatting, while ArcQuant (Meng et al., 2026) preserves information with residual activation channels. FAAR (Li et al., 2026) is closely related but adapts the *rounding* behavior to the non-uniform NVFP4 grid. H-Scale is complementary because it keeps the hardware-native rounding rule fixed and optimizes the *per-group scales*, serving as a **baseline-agnostic refinement for NVFP4 pipelines** and a drop-in enhancement for Blackwell-generation inference engines.

Scale Refinement for LLM Quantization. The critical role of scaling factors in low-bit quantization has motivated a line of work on *refining* or *optimizing* scales beyond naive min-max or per-group max normalization. LSQ (Esser et al., 2020) learns quantization scales via backpropagation during fine-tuning, but its training cost is prohibitive for modern LLMs. SmoothQuant (Xiao et al., 2023) migrates activation outliers into weights via layer-wise scale migration to ease symmetric quantization. It focuses on scale placement between activations and weights rather than post-quantization scale tuning. Similar to our approach, SignRoundV2 (Cheng et al., 2025) performs lightweight search over quantization scales before fine-tuning the weights. MR-GPTQ (Egiazarian et al., 2025) likewise refines scales prior to applying GPTQ in the NVFP4 setting.

H-Scale aligns with this scale-refinement philosophy but differs in two aspects: (1) the objective is a diagonal second-order, output-aware reconstruction loss, and (2) the intervention sits at the final per-group scale-selection step of the NVFP4 pipeline, so the method is complementary to prior reconstruction or structural quantization.

3 METHOD

In this section, we formulate NVFP4 quantization and introduce H-Scale as a per-group scale refinement method. We first replace plain weight MSE with a diagonal-Hessian weighted reconstruction objective, then choose each group's scale by enumerating a short window of neighboring E4M3-representable local scales.

3.1 PRELIMINARIES: NVFP4 QUANTIZATION

The NVFP4 format optimizes the trade-off between hardware efficiency and expressivity via a micro-block design. We denote the *original* full-precision weight matrix by $\mathbf{W} \in \mathbb{R}^{m\times n}$ and the *quantized* weight matrix by $\hat{\mathbf{W}}$. Each row of $\mathbf{W}$ is partitioned into contiguous groups of size $g = 16$ along the input-channel dimension. We write $\mathbf{w}_{r,i} \in \mathbb{R}^g$ for the i-th group in row r, and $\hat{\mathbf{w}}_{r,i}$ for its quantized counterpart.

In hardware, NVFP4 represents each group using a 4-bit E2M1 value grid together with a per-group FP8 E4M3 local scale and a tensor-level FP32 global scale. We write $\ell_{r,i}$ for the local scale and s_{g} for the global scale, and define the *effective group scale*

$$s_{r,i} = s_{\mathrm{g}}\, \ell_{r,i} \tag{1}$$

as the scalar actually used to dequantize the group. Formally, the standard quantization function for a group $\mathbf{w}_{r,i}$ is:

$$\hat{\mathbf{w}}_{r,i} = s_{r,i} \cdot \mathrm{Clip}\left(\mathrm{Round}\left(\frac{\mathbf{w}_{r,i}}{s_{r,i}}\right)\right), \tag{2}$$

where $\mathrm{Round}(\cdot)$ maps values to the nearest FP4 discrete levels and $\mathrm{Clip}(\cdot)$ bounds the result within the FP4 representable range $[-6.0, 6.0]$. Conventionally, $s_{r,i}$ is initialized from the maximum absolute value within the group and then snapped to the hardware-valid scale representation.

3.2 HESSIAN-GUIDED SCALE OBJECTIVE

We start from the *layer output reconstruction* objective rather than plain weight MSE. Let $\mathbf{W} \in \mathbb{R}^{m\times n}$ be a weight matrix, $\hat{\mathbf{W}}$ its quantized counterpart, and $\Delta\mathbf{W} = \mathbf{W} - \hat{\mathbf{W}}$. Given a calibration activation matrix $\mathbf{X} \in \mathbb{R}^{N\times n}$, the output mismatch over the calibration set is

$$\|\mathbf{X}\mathbf{W}^\top - \mathbf{X}\hat{\mathbf{W}}^\top\|_F^2 = \|\mathbf{X}\Delta\mathbf{W}^\top\|_F^2 = \mathrm{Tr}\big(\Delta\mathbf{W}\,\mathbf{X}^\top\mathbf{X}\,\Delta\mathbf{W}^\top\big)\,. \tag{3}$$

This objective involves the full second-order weighting matrix $\mathbf{G} = \mathbf{X}^\top\mathbf{X}$, which is the Hessian of the squared output reconstruction loss. Directly optimizing it over all NVFP4 group scales would couple the scale choices across input-channel groups through the off-diagonal entries of $\mathbf{G}$, making the exact discrete search combinatorial. We therefore use a diagonal-Hessian surrogate by retaining only

$$\mathbf{h} = \mathrm{diag}(\mathbf{X}^\top\mathbf{X}) \in \mathbb{R}^n, \tag{4}$$

where each h_j measures the activation energy of input channel j on the calibration set. Under this diagonal approximation, the objective decouples across output rows and input-channel groups, yielding a weighted reconstruction loss:

$$\mathrm{Tr}\big(\Delta\mathbf{W}\,\mathbf{X}^\top\mathbf{X}\,\Delta\mathbf{W}^\top\big) \approx \sum_{r=1}^{m}\sum_{j=1}^{n} h_j\,(\Delta W_{rj})^2. \tag{5}$$

The approximation drops the cross-channel correlation terms. High-energy input channels are still penalized more heavily, because they contribute more strongly to $\mathbf{X}\Delta\mathbf{W}^\top$, while weakly activated channels have less influence on the layer output. The scale search therefore spends the limited FP4 resolution on channels that dominate the calibration output error.

For NVFP4, the diagonal Hessian proxy must respect the same group partition. Let $\mathbf{h}_i \in \mathbb{R}^g$ be the slice of $\mathbf{h}$ aligned with group index i. For each output row r and group i, we select the effective group scale from the hardware-valid candidate set by minimizing the corresponding weighted reconstruction error:

$$\min_{s_{r,i}\in\mathcal{S}_{r,i}} \left\| (\mathbf{w}_{r,i} - \hat{\mathbf{w}}_{r,i}(s_{r,i})) \odot \sqrt{\mathbf{h}_i} \right\|_2^2, \tag{6}$$

where $\odot$ denotes element-wise multiplication and $\mathcal{S}_{r,i}$ denotes the finite set of hardware-valid *effective* scales induced by a length-K=16 bidirectional walk on the E4M3 local-scale grid, constructed in the next subsection. In practice, we compute $\mathbf{h}$ once per layer from the calibration activations and slice it according to the same input-channel groups used by NVFP4. Eq. (6) is a group-wise diagonal surrogate of Eq. (3): it keeps the per-channel Hessian weights and drops cross-channel terms. Given $\mathcal{S}_{r,i}$, we evaluate every candidate and keep the minimizer, and the search is exhaustive over those K scales. When the group weights are replaced by the all-ones vector $\mathbf{h}_i = \mathbf{1}$, Eq. (6) degenerates to the plain weight-centric search objective used in our ablations.

3.3 Efficient Scale Optimization Algorithm

Optimizing Eq. (6) is difficult because both the FP4 projection and the scale representation are discrete. Our implementation therefore uses a deterministic *one-dimensional scale search* that preserves the original NVFP4 quantization pipeline, i.e., we keep the baseline grouping and value grid unchanged, and only refine the per-group scale.

Initialization from the baseline quantizer. For each group (r, i), we first obtain an initial effective scale $s^{\text{init}}_{r,i}$ from the underlying quantizer. In the plain RTN variant, this is the usual max-abs scale induced by the group maximum together with the FP8 scale quantization step. H-Scale does not alter this initialization rule and only refines the resulting scale.

Hardware-valid scale candidate set. Starting from $s^{\text{init}}_{r,i}$, H-Scale evaluates that scale and a few neighboring E4M3-representable scales on both sides. The two directions help for different reasons, because E2M1 is a non-uniform grid on $[-6, 6]$: a smaller scale stretches $\mathbf{w}/s$ toward ± 6, so outliers may clip while the remaining weights occupy more of the grid. A larger scale compresses $\mathbf{w}/s$ toward zero, so the group drops the coarsest codes $\{\pm 4, \pm 6\}$ and large-magnitude weights are rounded on denser mid-range codes. Which side wins is decided by the Hessian-weighted error in Eq. (6), not by fitting the group range.

Because $s_{r,i} = s_{\text{g}}\, \ell_{r,i}$ as in Eq. (1), it is enough to walk on the E4M3 local-scale grid and multiply back by the shared FP32 global scale. Let $Q_{\text{fp8}}(\cdot)$ denote FP8 E4M3 quantization. The initialized local scale is

$$\ell^{\text{init}}_{r,i} = Q_{\text{fp8}}\left(\frac{s^{\text{init}}_{r,i}}{s_{\text{g}}}\right), \tag{7}$$

with corresponding effective scale $s_{\text{g}}\, \ell^{\text{init}}_{r,i}$. Thus only $\ell_{r,i}$ is constrained to the hardware FP8 grid, while s_{g} remains in FP32.

Let

$$\mathcal{L}^{+} = (\lambda_1 < \lambda_2 < \cdots < \lambda_M) \tag{8}$$

be the sorted positive finite E4M3 values, and let $\iota(\ell)$ be the index of ℓ in this list. From $\ell^{\text{init}}_{r,i}$, we take the D nearest smaller values, the initialization itself, and the U nearest larger values:

$$\mathcal{D} = \{-D, \ldots, -1,\ 0,\ +1, \ldots, +U\}, \qquad D = K - 1 - U, \qquad |\mathcal{D}| = K, \tag{9}$$

which defines the *local-scale* candidate set

$$\mathcal{L}_{r,i} = \Big\{ \lambda_{\,\text{clip}\left(\,\iota(\ell^{\text{init}}_{r,i})+\delta,\ 1,\ M\,\right)} \;\Big|\; \delta \in \mathcal{D} \Big\}. \tag{10}$$

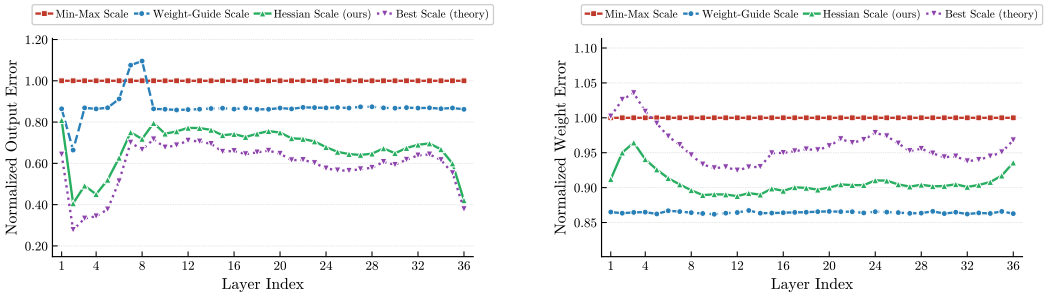
Thus $\delta > 0$ selects a larger scale, $\delta < 0$ a smaller one, and $\delta = 0$ recovers the initialization. Throughout the paper we use $U = 6$ and $K = 16$, i.e. nine steps down and six steps up. Shifts that would leave the grid are clipped to the endpoints. Every candidate is exactly representable in E4M3 by construction.

Mapping each local scale back through the global factor yields the *effective-scale* candidate set used in Eq. (6):

$$\mathcal{S}_{r,i} = \big\{\, s_{\text{g}}\, \ell \;\big|\; \ell \in \mathcal{L}_{r,i} \,\big\}. \tag{11}$$

Each candidate $\tilde{s} \in \mathcal{S}_{r,i}$ is evaluated by standard FP4 group quantization,

$$\hat{\mathbf{w}}_{r,i}(\tilde{s}) = Q_{\text{fp4}}\Big(\frac{\mathbf{w}_{r,i}}{\tilde{s}}\Big)\, \tilde{s}, \tag{12}$$



(a) Normalized layer output error

(b) Normalized weight MSE

Figure 1: **Gate-projection layer-wise validation on Qwen3-4B-Instruct.** Both metrics are normalized by the Min-Max baseline at each layer. Although Weight-Guided Scale gives the lowest unweighted weight MSE, H-Scale closely tracks the Best Scale oracle for layer output error, demonstrating the mismatch between weight reconstruction and output preservation.

and the best hardware-valid scale is chosen independently for each group:

$$\tilde{s}^*_{r,i} = \arg\min_{\tilde{s}\in\mathcal{S}_{r,i}} \left\| (\mathbf{w}_{r,i} - \hat{\mathbf{w}}_{r,i}(\tilde{s})) \odot \sqrt{\mathbf{h}_i} \right\|_2^2. \tag{13}$$

When $\mathbf{h}_i = \mathbf{1}$, the same procedure reduces to the scale-only search in our ablations.

Final re-quantization. After the best hardware-valid scale $\tilde{s}^*_{r,i}$ is chosen, we perform one final FP4 quantization with that scale. The search is offline and fully separable: different groups, and the K candidates inside each group, can be scored independently, so the inner loop batches on GPU as FP4 projection plus element-wise weighted error. Relative to weight-update methods such as GPTQ, the extra cost is small, and inference is unchanged. This is why H-Scale can sit as a drop-in refinement on RTN, 4over6, and other NVFP4 pipelines.

3.4 Why Hessian-Guided Scale Search Works

Traditional NVFP4 RTN treats the group scale as a range-fitting parameter. For each 1×16 weight group, it chooses a min-max (or max-abs) scale so that all values fall within the representable FP4 range, and then rounds every weight to the induced E2M1 grid. Min-max scaling only fits that range. It ignores both the rounding of the coarse FP4 grid and the distribution of the remaining weights.

A single outlier can therefore determine the scale for all 16 values and decide which FP4 codes are actually used by the majority of weights. Scale-refinement methods such as SignRoundV2 (Cheng et al., 2025) and MR-GPTQ (Egiazarian et al., 2025) improve on this by searching the initial scale to minimize $\|\mathbf{w} - \hat{\mathbf{w}}\|_2^2$. This is exactly a special case of Eq. (6) with $\mathbf{h}_i = \mathbf{1}$, i.e., every coordinate in the group is treated as equally important. H-Scale keeps the same hardware-valid NVFP4 scale space, but replaces this identity weighting with the diagonal Hessian proxy. As a result, the chosen scale allocates FP4 resolution toward weights that matter most for the calibration-layer output.

Figure 1 evaluates this distinction on all 36 `gate_proj` layers of Qwen3-4B-Instruct, with every curve normalized by the Min-Max baseline at the same layer (red line $\equiv 1.0$). This validation uses 4096 FineWeb (Penedo et al., 2024) calibration samples with sequence length 8192. We report both layer-output error and unweighted weight MSE to test whether the diagonal-Hessian objective in Eq. (6) is a better surrogate for Eq. (3) than plain reconstruction error.

Min-max scaling is the standard RTN range-fitting rule above. *Weight-centric scale search* ("Weight-Guided Scale" in the plot) searches the same scale candidates as H-Scale, but minimizes unweighted $\|\mathbf{w}-\hat{\mathbf{w}}\|_2^2$ by setting $\mathbf{h}_i = \mathbf{1}$. *Best Scale* is an oracle reference obtained by backpropagating through the scale variables to optimize the full layer-output objective in Eq. (3).

Relative to Min-Max, H-Scale reduces activation loss by 32.5% on average, close to the 40.8% reduction of Best Scale (about 80% of the oracle gain). Best Scale fine-tunes continuous scale variables by gradient descent and is much more expensive than the discrete search used at PTQ time.

Table 1: Performance evaluation on Qwen3-30A3-Thinking. H-Scale generally improves average performance across PTQ pipelines, though not every individual benchmark improves for every method.

Method	Avg.	AIME24	AIME25	MMLU-R	LiveBench	LCBench
BF16 Baseline	81.06	89.52	78.10	90.85	76.23	70.58
RTN	80.03	86.76	77.61	90.54	**75.14**	70.12
+ H-Scale	**80.68**	**89.33**	**77.98**	**90.65**	74.72	**70.74**
4over6	79.99	87.44	77.04	90.23	75.04	70.20
+ H-Scale	**80.49**	**87.82**	**77.84**	**90.72**	**75.53**	**70.56**
ArcQuant	79.73	87.91	74.53	90.42	**75.74**	**70.04**
+ H-Scale	**80.46**	**89.03**	**77.78**	**90.89**	74.65	69.97
GPTQ	80.01	86.67	78.76	**90.77**	74.16	69.71
+ H-Scale	**81.22**	**88.94**	**80.45**	90.07	**76.13**	**70.49**
GPTAQ	80.21	87.63	78.32	**90.79**	74.33	**69.99**
+ H-Scale	**80.71**	**88.42**	**78.65**	90.71	**76.09**	69.69
MR-GPTQ	79.67	85.35	**78.00**	**90.81**	75.25	68.93
+ H-Scale	**80.41**	**87.00**	77.95	90.59	**76.03**	**70.49**

Weight-Guided Scale attains the lowest unweighted weight MSE, yet reduces activation loss by only 12.6%. Therefore, H-Scale approaches the oracle activation-loss reduction through a lightweight Hessian-guided search.

3.5 Integration with NVFP4 SOTA: H-Scale as an RTN Replacement

Most NVFP4 pipelines end with the same local step: pick a per-group scale, typically by max-abs / RTN, and round the group to the E2M1 grid. H-Scale replaces only that pick. Thus, **any NVFP4 pipeline whose final local projection uses an RTN-style scale selector can replace that selector with H-Scale** without changing the rest of the algorithm. This yields a unified view of how H-Scale plugs into existing SOTA.

Methods such as 4over6 (Cook et al., 2025) and ArcQuant (Meng et al., 2026) first optimize structure (e.g., 4/6 mapping, residual channels), then perform per-group quantization. H-Scale leaves those structural decisions intact and replaces the final local scale selection: start from the baseline's chosen scale, run the Hessian-weighted search over the effective-scale candidates in Eq. (11), and re-quantize with the selected hardware-valid effective scale.

GPTQ (Frantar et al., 2023) and variants (GPTAQ, MR-GPTQ) use the full Hessian $\mathbf{H}$ for column ordering and for propagating rounding error across columns via $\mathbf{H}^{-1}$. The local projection to the NVFP4 grid, however, still requires choosing a group scale and rounding to the grid. H-Scale replaces this local scale-selection step: for each group, it uses $\mathbf{h} = \mathrm{diag}(\mathbf{H})$ in the scale-search objective Eq. (6). GPTQ keeps its ordering and error propagation, and only the scale selection becomes Hessian-guided. H-Scale is therefore a drop-in replacement for the RTN-style scale-selection step inside these pipelines.

A single calibration pass yields $\mathbf{h}$ per layer, or reuses $\mathrm{diag}(\mathbf{H})$ when GPTQ already built it. The same replacement then applies to RTN, 4over6, ArcQuant, and GPTQ-like methods. The NVFP4 format and inference cost do not change.

4 Experiments

Our experiments ask whether H-Scale raises the average performance of strong NVFP4 baselines, transfers across model scales and families, and provides enough benefit to justify its one-time offline Hessian cost.

Table 2: Evaluation on Qwen3-30A3-Instruct. H-Scale usually improves average accuracy across eight diverse tasks, with mixed effects on individual benchmarks.

Method	Avg.	C-Eval	LiveBench	MMLU-R	AIME25	LCBench	ARC-C	BBH	GPQA
BF16 Baseline	65.58	63.43	69.27	89.11	59.90	48.42	69.05	70.50	54.99
RTN	64.04	62.51	**68.34**	**89.23**	56.91	47.56	67.35	**68.33**	52.07
+ H-Scale	**64.78**	**62.73**	68.23	89.14	**57.69**	**47.61**	**68.10**	67.47	**57.24**
4over6	64.25	62.44	68.04	88.47	**57.88**	47.63	**68.50**	67.96	53.11
+ H-Scale	**64.65**	**62.77**	**68.49**	**89.19**	57.65	**48.19**	68.37	**68.03**	**54.54**
ArcQuant	63.45	62.59	66.82	**89.02**	55.03	46.55	**68.16**	67.45	52.00
+ H-Scale	**64.24**	**63.03**	**67.24**	88.70	**55.68**	**47.13**	67.58	**67.49**	**57.04**
GPTQ	63.90	**63.29**	68.84	**88.74**	56.22	**48.43**	67.31	66.98	51.38
+ H-Scale	**64.50**	62.45	**69.29**	88.59	**56.37**	47.44	**68.40**	**68.38**	**55.11**
GPTAQ	64.27	**62.65**	**69.12**	**89.22**	**58.55**	47.22	67.61	**67.70**	52.11
+ H-Scale	**64.74**	62.62	69.10	88.82	58.02	**48.68**	**68.68**	67.38	**54.61**
MR-GPTQ	63.63	62.28	**68.02**	88.53	55.68	47.68	**68.39**	**68.15**	50.34
+ H-Scale	**64.48**	**62.66**	67.41	**88.75**	**59.85**	**47.82**	68.07	67.29	**54.00**

Table 3: Evaluation on Qwen3-4B-Instruct. H-Scale yields higher observed average scores across the tested NVFP4 pipelines, while individual benchmark gains remain method-dependent.

Method	Avg.	C-Eval	LiveBench	MMLU-R	AIME25	LCBench	ARC-C	BBH	GPQA
BF16 Baseline	57.76	55.54	63.16	83.98	46.16	36.63	60.72	73.50	42.41
RTN	55.38	**52.48**	60.66	82.87	40.37	34.32	60.28	71.76	40.29
+ H-Scale	**56.26**	52.43	**61.08**	**83.13**	**43.24**	**35.65**	**60.69**	**72.39**	**41.44**
4over6	55.38	52.20	**60.43**	**82.74**	43.35	**36.16**	**60.10**	70.16	37.91
+ H-Scale	**56.12**	**54.70**	59.95	82.73	**43.51**	35.46	59.30	**72.40**	**40.95**
ArcQuant	54.49	52.42	58.87	81.27	40.19	**33.80**	58.41	70.58	**40.42**
+ H-Scale	**55.00**	**53.46**	**59.63**	**81.39**	**42.97**	33.06	**58.56**	**71.26**	39.63
GPTQ	55.63	52.42	**61.86**	82.51	**42.18**	**35.24**	59.87	68.17	42.78
+ H-Scale	**56.53**	**55.11**	60.39	**82.87**	42.07	34.70	**60.29**	**71.23**	**45.61**
GPTAQ	56.63	53.13	**62.19**	**83.44**	**45.72**	**36.47**	60.06	71.00	41.02
+ H-Scale	**57.52**	**54.17**	61.68	83.07	43.77	36.17	**60.68**	**73.05**	**47.54**
MR-GPTQ	56.41	53.54	62.57	**83.16**	44.28	**35.13**	60.33	70.89	41.38
+ H-Scale	**56.74**	**54.86**	**62.84**	82.38	**44.50**	34.52	**60.84**	**71.30**	**42.71**

4.1 Experimental Setup

Benchmarks. We use AIME, MMLU-Redux, LiveBench, LiveCodeBench, C-Eval, ARC-Challenge (ARC-C), BBH, GPQA, and GSM8K to evaluate reasoning, math, knowledge, and coding ability (Clark et al., 2018; Cobbe et al., 2021; Gema et al., 2025; Huang et al., 2023; Jain et al., 2025; Mathematical Association of America, 2024; Rein et al., 2024; Suzgun et al., 2023; White et al., 2025). Model-specific benchmark suites are detailed in Appendix I.

Decoding. All methods use a fixed decoding protocol for paired comparisons. Because generation is stochastic, we evaluate each setting with **three** independent decoding runs and report the mean score. Complete generation settings are provided in Appendix I.

Baselines. H-Scale is applied to NVFP4 weight-only PTQ pipelines at the hardware-native group size g=16: RTN, GPTQ (Frantar et al., 2023), 4over6 (Cook et al., 2025), ArcQuant (Meng et al., 2026), MR-GPTQ (Egiazarian et al., 2025), and GPTAQ (Li et al., 2025). All methods use 4096 FineWeb (Penedo et al., 2024) calibration samples with sequence length 8192. We do not include activation-migration methods such as SmoothQuant (Xiao et al., 2023) or AWQ (Lin et al., 2024), or Hadamard-based preprocessing: g=16 is fixed by Blackwell tensor-core support, and prior NVFP4 work finds that per-group scales already handle local outliers sufficiently well that these redistribution techniques yield little additional benefit (Egiazarian et al., 2025).

Table 4: Cross-family evaluation on LLaMA-3.1-8B-Instruct. Avg. is computed over the seven listed benchmarks. H-Scale yields higher observed average scores for all tested NVFP4 baselines.

Method	Avg.	C-Eval	LiveBench	MMLU-R	ARC-C	GSM8K	BBH	GPQA
BF16 Baseline	53.02	51.05	19.24	66.91	51.83	76.89	68.24	36.97
RTN	48.79	45.64	18.17	63.07	49.08	70.92	**62.41**	**32.21**
+ H-Scale	**49.53**	**48.15**	**18.77**	**64.30**	**49.70**	**72.52**	61.43	31.84
4over6	48.86	46.70	18.63	61.68	49.73	72.19	62.24	30.87
+ H-Scale	**50.05**	**48.21**	**19.16**	**63.16**	**50.21**	**74.28**	**64.02**	**31.31**
ArcQuant	48.40	46.30	17.92	**62.74**	49.94	**72.84**	60.32	**28.72**
+ H-Scale	**48.64**	**49.19**	**18.23**	62.01	**51.06**	71.25	**61.09**	27.64
GPTQ	50.61	48.93	19.20	**65.32**	51.73	70.91	63.57	34.63
+ H-Scale	**51.32**	**50.28**	**19.45**	63.00	**51.95**	**75.33**	**64.08**	**35.17**
GPTAQ	49.54	46.02	17.82	64.49	50.20	70.29	**64.22**	33.74
+ H-Scale	**51.48**	**50.61**	**19.73**	**65.05**	**50.84**	**73.74**	63.75	**36.65**
MR-GPTQ	49.10	47.94	16.93	64.23	**51.78**	67.31	63.88	31.60
+ H-Scale	**50.11**	**48.60**	**18.60**	**64.31**	50.20	**68.76**	**64.28**	**36.04**

Table 5: Ablation on Qwen3-30A3-Instruct: scale with vs. without Hessian. "Scale-only" uses identity weighting in Eq. (6).

Baseline	Variant	Avg.	C-Eval	LiveBench	MMLU-R	AIME25	LCBench	ARC-C	BBH	GPQA
4over6	Baseline	64.25	62.44	68.04	88.47	**57.88**	47.63	**68.50**	67.96	53.11
	Scale-only (w/o $\mathbf{H}$)	63.55	**64.92**	66.65	87.33	56.58	46.72	67.15	66.53	52.52
	+ H-Scale	**64.65**	62.77	**68.49**	**89.19**	57.65	**48.19**	68.37	**68.03**	**54.54**
ArcQuant	Baseline	63.45	62.59	66.82	**89.02**	55.03	46.55	**68.16**	67.45	52.00
	Scale-only (w/o $\mathbf{H}$)	63.10	62.98	66.55	87.54	54.58	47.06	66.81	66.80	52.51
	+ H-Scale	**64.24**	**63.03**	**67.24**	88.70	**55.68**	**47.13**	67.58	**67.49**	**57.04**
GPTQ	Baseline	63.90	**63.29**	68.84	88.74	56.22	**48.43**	67.31	66.98	51.38
	Scale-only (w/o $\mathbf{H}$)	64.39	62.76	67.51	**89.24**	**56.67**	47.92	**68.43**	67.52	55.05
	+ H-Scale	**64.50**	62.45	**69.29**	88.59	56.37	47.44	68.40	**68.38**	**55.11**

4.2 Main Results

Large Reasoning Model. Table 1 shows that H-Scale yields a higher average score for every PTQ pipeline on Qwen3-30A3-Thinking. H-Scale also narrows the gap to the BF16 baseline (81.06) for every pipeline, with GPTQ + H-Scale reaching **81.22**. Observed gains are most consistent on AIME24 and AIME25, suggesting that output-aware scale alignment helps preserve reasoning behavior in NVFP4.

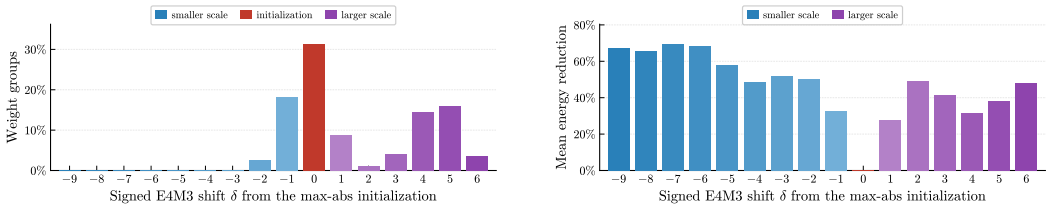
Instruction-Tuned Model. Table 2 reports the eight-task suite for Qwen3-30A3-Instruct. H-Scale again yields higher averages for every baseline.

Smaller Model Scaling. Table 3 extends the evaluation to Qwen3-4B-Instruct. Improvements are visible on GPQA, BBH, and AIME25, indicating that Hessian-guided scale refinement remains useful at lower model capacity.

Cross-Family Transfer. Table 4 evaluates LLaMA-3.1-8B-Instruct to test whether the gains are Qwen-specific. H-Scale yields higher average scores for every tested baseline, with especially consistent gains on C-Eval and LiveBench.

4.3 Ablation Analysis

Is the Hessian necessary? We compare the raw baseline, a *scale-only* variant that replaces the Hessian diagonal in Eq. (6) with the identity, and full *H-Scale*. Table 5 reports the Qwen3-30A3-Instruct ablation. H-Scale gives the highest average score for all three baselines, while scale-only optimization is less stable and can hurt 4over6 and ArcQuant.



(a) Selected shift δ (b) Mean weighted-error reduction

Figure 2: **Scale shifts on Qwen3-4B-Instruct.** Across all Qwen3-4B-Instruct MLP weight groups: (a) selected E4M3 shifts from max-abs initialization. (b) mean per-group loss reduction of Eq. (6) relative to initialization. Positive δ enlarges the scale.

Table 6: Candidate-window sweep on all linear layers of Qwen3-4B-Instruct. Median and 10th percentile (P10) of the recovered improvement in layer-output error from max-abs to full E4M3 search (%). Default in bold.

(U, K)	$(0, 16)$	$(4, 16)$	$(5, 16)$	$(\mathbf{6}, \mathbf{16})$	$(7, 16)$	$(8, 16)$	$(10, 16)$	$(6, 20)$	$(6, 24)$
Median (%)	64.63	90.29	98.59	**100.00**	100.00	100.00	100.00	100.00	100.00
P10 (%)	57.94	87.46	97.83	**99.91**	99.89	99.84	99.63	99.96	99.96

Selected scale shifts. Figure 2 pools all MLP projections of Qwen3-4B-Instruct. Only 31.2% of groups retain the max-abs scale. 47.9% select larger scales and 20.9% smaller ones, supporting bidirectional search. Shift frequency alone does not measure its contribution: $\delta = -1$ reduces the Hessian-weighted error by 32.8% on average, whereas deep contractions ($\delta \leq -4$) reduce it by 58.7%. Although the latter involve only 0.11% of groups, they account for 8.3% of the total error removed. Overall, H-Scale removes 50.6% of the initialization's weighted error.

Candidate-window width. Table 6 varies the upward budget U and total budget K, with $D = K - 1 - U$ downward steps. We compare each window with a reference search over all positive finite E4M3 scales; both use the diagonal objective in Eq. (6) to select scales. Let L_{init}, L_{window}, and L_{grid} denote the full layer-output error in Eq. (3) for the max-abs initialization, window search, and full-grid search, respectively. We report $(L_{\text{init}} - L_{\text{window}})/(L_{\text{init}} - L_{\text{grid}})$, the fraction of the full-grid improvement recovered by each window. The window $(U, K) = (6, 16)$ reaches 100.00% at the median and 99.91% at P10. Increasing K to 20 or 24 improves P10 only to 99.96%. For convenience, we use $U = 6$ and $K = 16$ across all experiments.

Efficiency and Hardware Deployment. Table 7 reports offline costs on one NVIDIA H200 GPU. GPTQ reuses its Hessian, so H-Scale adds only scale search (9.6 to 9.8 h). 4over6 requires an additional calibration pass (0.2 to 0.6 h). These one-time costs leave the NVFP4 representation and inference cost unchanged.

Table 7: Offline runtime with matched hardware and calibration data.

Method	**Runtime**	**Method**	**Runtime**
4over6	0.2 h	GPTQ	9.6 h
4over6 + H-Scale	0.6 h	GPTQ + H-Scale	9.8 h

5 Conclusion

We identify per-group scale selection as an important bottleneck for NVFP4 fidelity and propose **H-Scale**, which selects hardware-valid scales using a diagonal-Hessian-weighted reconstruction objective. This lightweight refinement preserves the native NVFP4 representation with zero additional inference cost. Across Qwen and LLaMA models, H-Scale generally yields higher average performance over diverse PTQ baselines, while layer-wise validation and ablations support the value of output-aware scale selection.

AI USE STATEMENT

In this work, we have not used generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, formulate mathematical claims, provide ingredients for proofs, write proofs, propose or refine hypotheses, design experiments, implement the proposed method, translate the manuscript, clean or reformat datasets, perform qualitative analysis, or interpret experimental results. Additionally, we used generative AI tools to edit the manuscript for readability and to assist with LaTeX formatting of tables and style-file migration. All AI-assisted text was reviewed by the authors, and all numerical results, algorithms, and claims were verified against our own experiments and implementations. We take responsibility for the final content of this work, including text, claims, and artifacts.

REFERENCES

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33*, volume 33, pp. 1877–1901, 2020.

Wenhua Cheng, Weiwei Zhang, Heng Guo, Haihao Shen, and Zaner Ma. SignRoundV2: Toward closing the performance gap in extremely low-bit post-training quantization for LLMs, 2025. URL https://arxiv.org/abs/2512.04746v2. Revised May 2026 (v2).

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.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Jack Cook, Junxian Guo, Guangxuan Xiao, Yujun Lin, Keith Wyss, Mahdi Nazemi, Asit Mishra, Carlo del Mundo, Tijmen Blankevoort, and Song Han. Four over six: More accurate NVFP4 quantization with adaptive block scaling. *arXiv preprint arXiv:2512.02010*, 2025. URL https://arxiv.org/abs/2512.02010v5. Revised May 2026 (v5).

Vage Egiazarian, Roberto L. Castro, Denis Kuznedelev, Andrei Panferov, Eldar Kurtic, Shubhra Pandit, Alexandre Marques, Mark Kurtz, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Bridging the gap between promise and performance for microscaling fp4 quantization, 2025. URL https://arxiv.org/abs/2509.23202.

Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. Learned step size quantization. In *International Conference on Learning Representations*, 2020.

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. URL https://arxiv.org/abs/2210.17323.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, Claire Barale, Robert McHardy, Joshua Harris, Jean Kaddour, Emile van Krieken, and Pasquale Minervini. Are we done with MMLU? 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. 5069–5096, Albuquerque, New Mexico, apr 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.naacl-long.262. URL https://aclanthology.org/2025.naacl-long.262/.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza,

Michiel Blokzijl, Steven Bohez, Konstantinos Bousmalis, Anthony Brohan, Thomas Buschmann, Arunkumar Byravan, Serkan Cabi, Ken Caluwaerts, Federico Casarini, Oscar Chang, Jose Enrique Chen, Xi Chen, Hao-Tien Lewis Chiang, Krzysztof Choromanski, David D'Ambrosio, Sudeep Dasari, Todor Davchev, Coline Devin, Norman Di Palo, Tianli Ding, Adil Dostmohamed, Danny Driess, Yilun Du, Debidatta Dwibedi, Michael Elabd, Claudio Fantacci, Cody Fong, Erik Frey, Chuyuan Fu, Marissa Giustina, Keerthana Gopalakrishnan, Laura Graesser, Leonard Hasenclever, Nicolas Heess, Brandon Hernaez, Alexander Herzog, R. Alex Hofer, Jan Humplik, Atil Iscen, Mithun George Jacob, Deepali Jain, Ryan Julian, Dmitry Kalashnikov, M. Emre Karagozler, Stefani Karp, Chase Kew, Jerad Kirkland, Sean Kirmani, Yuheng Kuang, Thomas Lampe, Antoine Laurens, Isabel Leal, Alex X. Lee, Tsang-Wei Edward Lee, Jacky Liang, Yixin Lin, Sharath Maddineni, Anirudha Majumdar, Assaf Hurwitz Michaely, Robert Moreno, Michael Neunert, Francesco Nori, Carolina Parada, Emilio Parisotto, Peter Pastor, Acorn Pooley, Kanishka Rao, Krista Reymann, Dorsa Sadigh, Stefano Saliceti, Pannag Sanketi, Pierre Sermanet, Dhruv Shah, Mohit Sharma, Kathryn Shea, Charles Shu, Vikas Sindhwani, Sumeet Singh, Radu Soricut, Jost Tobias Springenberg, Rachel Sterneck, Razvan Surdulescu, Jie Tan, Jonathan Tompson, Vincent Vanhoucke, Jake Varley, Grace Vesom, Giulia Vezzani, Oriol Vinyals, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Fei Xia, Ted Xiao, Annie Xie, Jinyu Xie, Peng Xu, Sichun Xu, Ying Xu, Zhuo Xu, Yuxiang Yang, Rui Yao, Sergey Yaroshenko, Wenhao Yu, Wentao Yuan, Jingwei Zhang, Tingnan Zhang, Allan Zhou, and Yuxiang Zhou. Gemini robotics: Bringing AI into the physical world, 2025. URL https://arxiv.org/abs/2503.20020.

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference, 2021. URL https://arxiv.org/abs/2103.13630.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. C-Eval: A multi-level multi-discipline chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 62991–63010, 2023. doi: 10.52202/075280-2749. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/c6ec1844bec96d6d32ae95ae694e23d8-Abstract-Datasets_and_Benchmarks.html.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida I. Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination-free evaluation of large language models for code. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

Hanglin Li, Shuchang Tian, Chen Lin, Zhiyong Zhao, and Kun Zhan. FAAR: Format-aware adaptive rounding for NVFP4. *arXiv preprint arXiv:2603.22370*, 2026.

Yuhang Li, Ruokai Yin, Donghyun Lee, Shiting Xiao, and Priyadarshini Panda. GPTAQ: Efficient finetuning-free quantization for asymmetric calibration. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 36690–36706. PMLR, 2025. URL https://proceedings.mlr.press/v267/li25dn.html.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for LLM compression and acceleration. In *MLSys*, 2024.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xucheng 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. doi: 10.1007/s11432-024-4235-6.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. SpinQuant: LLM quantization with learned rotations. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ogO6DGE6FZ.

Ahmed Masry, Xuan Long Do, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 2263–2279, 2022. URL https://aclanthology.org/2022.findings-acl.177.

Mathematical Association of America. American invitational mathematics examination (AIME), 2024. URL https://maa.org/math-competitions/american-invitational-mathematics-examination-aime.

Haoqian Meng, Yilun Luo, Yafei Zhao, Wenyuan Liu, Peng Zhang, and Xindian Ma. ARCQuant: Boosting NVFP4 quantization with augmented residual channels for LLMs. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8609–8623. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.388. URL https://aclanthology.org/2026.acl-long.388/.

NVIDIA. NVIDIA RTX Blackwell gpu architecture: Built for neural rendering. Technical whitepaper, NVIDIA Corporation, Santa Clara, CA, 2025. URL https://images.nvidia.com/aem-dam/Solutions/geforce/blackwell/nvidia-rtx-blackwell-gpu-architecture.pdf. Version 1.1.

Andrei Panferov, Erik Schultheis, Soroush Tabesh, and Dan Alistarh. Quartet II: Accurate LLM pretraining in NVFP4 by improved unbiased gradient estimation, 2026. URL https://arxiv.org/abs/2601.22813.

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. In *Advances in Neural Information Processing Systems 37*, 2024. URL https://arxiv.org/abs/2406.17557.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Anselm Levskaya, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. In *Proceedings of Machine Learning and Systems*, volume 5, pp. 606–624, 2023. URL https://proceedings.mlsys.org/paper_files/paper/2023/hash/c4be71ab8d24cdfb45e3d06dbfca2780-Abstract-mlsys2023.html.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://arxiv.org/abs/2507.02833.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026. URL https://qwen.ai/blog?id=qwen3.8.

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.

Bita Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, Dusan Stosic, Venmugil Elango, Maximilian Golub, Alexander Heinecke, Phil James-Roxby, Dharmesh Jani, Gaurav Kolhe, Martin Langhammer, Ada Li, Levi Melnick, Maral Mesmakhosroshahi, Andres Rodriguez, Michael Schulte, Rasoul Shafipour, Lei Shao, Michael Siu, Pradeep Dubey, Paulius Micikevicius, Maxim Naumov, Colin Verrilli, Ralph Wittig, Doug Burger, and Eric Chung. Microscaling data formats for deep learning, 2023. URL https://arxiv.org/abs/2310.10537.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. 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, 2023. URL https://aclanthology.org/2023.findings-acl.824.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with MATH-vision dataset. In *Advances in Neural Information Processing Systems*, volume 37, pp. 95095–95169, 2024. doi: 10.52202/079017-3014. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ad0edc7d5fa1a783f063646968b7315b-Abstract-Datasets_and_Benchmarks_Track.html.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh Agrawal, Sandeep Singh Sandha, Siddartha Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. Livebench: A challenging, contamination-limited LLM benchmark. In *International Conference on Learning Representations*, 2025. Spotlight.

xAI. RealWorldQA: A benchmark for real-world spatial understanding, 2024. URL https://huggingface.co/datasets/xai-org/RealworldQA.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pp. 38087–38099. PMLR, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

A Run-to-Run Variability of the Main Results

Following the protocol of Appendix I, every configuration is evaluated three times and the main-text tables report the per-run mean. This section reports the matching dispersion for the two Qwen3-30A3 suites: Table 8 for Table 1 and Table 9 for Table 2. Run k of a given configuration is a single end-to-end evaluation of all tasks, so the **Avg.** standard deviation is taken over the three per-run averages rather than propagated from the per-task columns.

Table 8: Run-to-run sample standard deviation (three runs, n=3, ddof=1) for every entry of Table 1 (Qwen3-30A3-Thinking, five-task suite). The **Avg.** column is the standard deviation of the three per-run averages, not a propagation of the per-task values.

Method	Avg.	AIME24	AIME25	MMLU-R	LiveBench	LCBench
BF16 Baseline	0.10	0.38	0.29	0.43	0.38	0.43
RTN	0.08	0.08	0.36	0.32	0.23	0.40
+ H-Scale	0.13	0.22	0.18	0.19	0.53	0.55
4over6	0.22	0.37	0.65	0.79	0.20	0.42
+ H-Scale	0.14	0.64	0.63	0.25	0.47	0.37
ArcQuant	0.14	0.21	0.45	0.19	0.47	0.11
+ H-Scale	0.19	0.80	0.43	0.57	0.22	0.33
GPTQ	0.37	0.26	0.36	0.17	0.61	0.59
+ H-Scale	0.10	0.14	0.46	0.28	0.16	0.44
GPTAQ	0.06	0.43	0.23	0.64	0.37	0.25
+ H-Scale	0.25	0.13	0.41	0.42	0.64	0.49
MR-GPTQ	0.20	0.26	0.70	0.32	0.33	0.65
+ H-Scale	0.08	0.41	0.35	0.51	0.40	0.71

Table 9: Run-to-run sample standard deviation (three runs, n=3, ddof=1) for every entry of Table 2 (Qwen3-30A3-Instruct, eight-task suite). The **Avg.** column is the standard deviation of the three per-run averages, not a propagation of the per-task values.

Method	Avg.	C-Eval	LiveBench	MMLU-R	AIME25	LCBench	ARC-C	BBH	GPQA
BF16 Baseline	0.10	0.10	0.47	0.37	0.63	0.54	0.38	0.25	0.49
RTN	0.10	0.39	0.51	0.20	0.41	0.57	0.53	0.35	0.46
+ H-Scale	0.27	0.40	0.51	0.54	0.30	0.28	0.41	1.00	0.57
4over6	0.13	0.34	0.22	0.43	0.24	0.47	0.29	0.31	0.41
+ H-Scale	0.25	0.09	0.48	0.53	0.42	0.57	0.59	0.22	0.19
ArcQuant	0.17	0.35	0.93	0.23	0.24	0.18	0.47	0.29	0.14
+ H-Scale	0.11	0.71	0.57	0.23	0.38	0.55	0.46	0.39	0.22
GPTQ	0.13	0.17	0.24	0.40	0.48	0.33	0.07	0.67	0.30
+ H-Scale	0.14	0.78	0.42	0.28	0.51	0.29	0.62	0.86	0.25
GPTAQ	0.05	0.33	0.29	0.46	0.23	0.46	0.58	0.54	0.29
+ H-Scale	0.16	0.47	0.31	0.18	0.17	0.42	0.27	0.34	0.42
MR-GPTQ	0.18	0.41	0.78	0.56	0.36	0.22	0.55	0.19	0.14
+ H-Scale	0.12	0.01	0.35	0.63	0.38	0.20	0.59	0.54	0.35

Per-task standard deviations range from 0.06 to 1.00 points with a median of 0.39, and at three runs no benchmark in either suite is systematically tighter than the others. Averaging over the suite cancels most of this jitter: the **Avg.** standard deviation never exceeds 0.37 points (median 0.14), the shrinkage expected from averaging five and eight tasks respectively. Those standard-deviation results demonstrate the effectiveness of our method: the suite-level fluctuation is small relative to the reported average gains, confirming that H-Scale improves overall model performance rather than reflecting run-to-run noise.

B HESSIAN ABLATION ON QWEN3-30A3-THINKING

Table 10 complements the ablation in the main text by repeating the same comparison on the Qwen3-30A3-Thinking benchmark suite. Scale-only optimization can recover part of the benefit over the raw baseline, but the Hessian-weighted objective gives the strongest average score for both ArcQuant and 4over6. This supports the interpretation that the gain is not merely from shrinking or retuning scales, but from weighting scale errors by their calibration-output sensitivity.

Table 10: Ablation on Qwen3-30A3-Thinking: scale with vs. without Hessian. "Scale-only" uses uniform unit weights in Eq. (6).

Baseline	Variant	Avg.	AIME24	AIME25	MMLU-R	LiveBench	LCBench
4over6	Baseline	79.99	87.44	77.04	90.23	75.04	70.20
	Scale-only (w/o $\mathbf{H}$)	80.05	87.71	76.67	90.39	**76.47**	69.03
	+ H-Scale	**80.49**	**87.82**	**77.84**	**90.72**	75.53	**70.56**
ArcQuant	Baseline	79.73	87.91	74.53	90.42	**75.74**	**70.04**
	Scale-only (w/o $\mathbf{H}$)	79.68	87.08	75.42	90.78	**75.74**	69.39
	+ H-Scale	**80.46**	**89.03**	**77.78**	**90.89**	74.65	69.97

C CALIBRATION-SET SIZE

To further verify the robustness of H-Scale, we reduce the calibration set from 4096 to 128 samples while keeping all other settings unchanged. Table 11 reports the result on Qwen3-30A3-Instruct: even with this much smaller budget, H-Scale still improves over the corresponding NVFP4 baseline.

Table 11: Qwen3-30A3-Instruct with a 128-sample calibration set.

Method	Avg.	C-Eval	LiveBench	MMLU-R	AIME25	LCBench	ARC-C	BBH	GPQA
BF16 Baseline	65.58	63.43	69.27	89.11	59.90	48.42	69.05	70.50	54.99
GPTQ	62.70	62.08	**66.10**	87.87	**56.25**	47.74	67.32	64.75	49.49
+ H-Scale	**63.32**	**62.97**	65.56	**87.90**	55.42	**48.14**	**67.89**	**66.15**	**52.53**
GPTAQ	63.44	62.75	66.31	**87.76**	54.33	48.21	67.41	66.20	54.55
+ H-Scale	**64.47**	**63.84**	**67.41**	87.47	**58.17**	48.21	**67.92**	**67.72**	**55.05**

D CALIBRATION MICROCASE

We first use a small calibration example to isolate why an output-aware scale objective can prefer a different FP4 code pattern from weight-centric search. Table 12 makes the objective mismatch explicit in a 4-dimensional calibration microcase with $\mathbf{w} = [-3.750, -1.740, 4.320, 4.880]$ and two calibration activations

$$\mathbf{X} = \begin{bmatrix} 0.600 & 1.400 & 1.500 & 2.000 \\ 2.000 & 1.000 & 0.700 & 1.000 \end{bmatrix}.$$

The output error is $\|\mathbf{X}(\mathbf{w} - \hat{\mathbf{w}})^\top\|_2^2$.

Table 12: A calibration microcase showing why scale search should be output-aware.

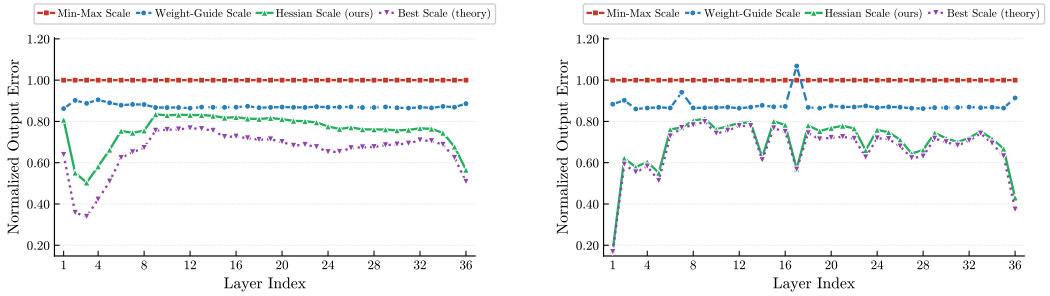
Scale choice	Objective	Scale	$\hat{\mathbf{w}}$	Weight / Output error
Min-max	Preserve range	0.813	$[-3.253, -1.627, 4.880, 4.880]$	0.573 / 3.927
Weight-centric search	$\min \|\mathbf{w} - \hat{\mathbf{w}}\|_2^2$	0.801	$[-3.204, -1.602, 4.805, 4.805]$	**0.559** / 3.448
Hessian-guided search	$\min \|\mathbf{X}(\mathbf{w} - \hat{\mathbf{w}})^\top\|_2^2$	0.895	$[-3.579, -1.789, 3.579, 5.368]$	0.820 / **0.079**

Min-max scaling is set by the largest-magnitude weight, so that coordinate fixes the grid for the whole group. Weight-centric search then finds the scale with the smallest unweighted reconstruction error, treating all four coordinates equally. Scoring candidates by layer-output error selects a

different FP4 code pattern, $[-4, -2, 4, 6]$, and a larger unweighted error (0.820 vs. 0.559). The output error falls from 3.927 (min-max) and 3.448 (weight-centric) to 0.079, because the residual is moved off the high-impact activation directions. H-Scale approximates this output-aware score with the diagonal Hessian proxy. The layer-wise plots below repeat the comparison at model scale.

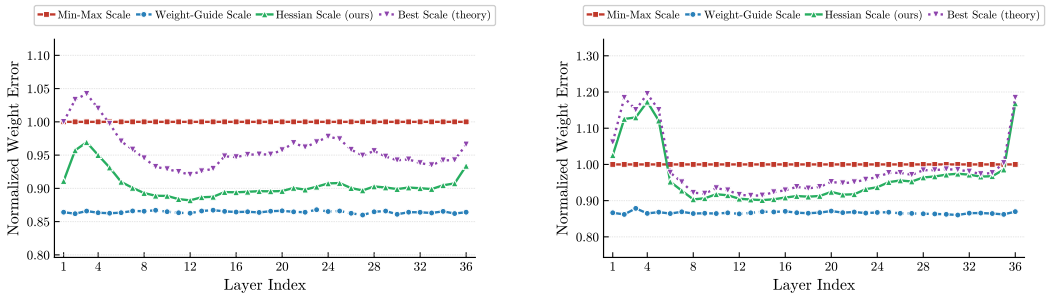
E Additional Layer-Wise Validation

The microcase above isolates the mechanism at the group level. We next check whether the same output-aware behavior appears consistently across layers. The layer-wise trend shown for `gate_proj` in Figure 1 also holds for the other MLP projections, so we report the additional `up_proj` and `down_proj` curves in Figures 3 and 4.



(a) `up_proj` output error (b) `down_proj` output error

Figure 3: **Additional normalized layer output error on Qwen3-4B-Instruct.** All curves are normalized by the Min-Max baseline at each layer. Best Scale is the oracle reference and gives the lowest output error, while H-Scale is the strongest practical scale-search method, reducing average error over Min-Max by 24% for `up_proj` and 30% for `down_proj`.



(a) `up_proj` weight MSE (b) `down_proj` weight MSE

Figure 4: **Additional normalized weight MSE on Qwen3-4B-Instruct.** Weight-Guided Scale generally minimizes unweighted reconstruction error, while H-Scale gives lower output error. For `down_proj`, H-Scale can exceed the Min-Max weight MSE at early layers and the final layer while still reducing output error, confirming that weight MSE is an unreliable proxy for layer output fidelity.

F Layer-Wise Validation with GPTQ Compensation

We next repeat the layer-wise comparison under GPTQ compensation for all 36 `up_proj` and `down_proj` layers of Qwen3-4B-Instruct, using the same calibration set as Figure 1.

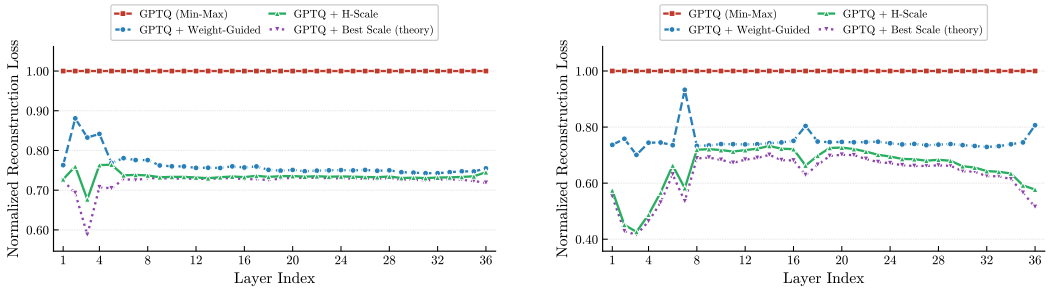
All methods use the groupwise GPTQ implementation in Algorithm 3: jointly project each 16-column group, then propagate its error to the remaining groups. There is no sequential compensation within the projected group. We fix the column order, tensor-level global scale, FP4 rounding rule,

and compensation procedure across methods. The baseline uses the E4M3-rounded max-abs scale. Weight-Guided Scale and H-Scale use the same $K = 16, U = 6$ candidate construction in Eq. (11), with unweighted and diagonal-Hessian-weighted scores, respectively. The purple reference, labeled "GPTQ + Best Scale (theory)" selects the candidate minimizing $\sum_j e_j(s)^2/U_{jj}^2$, where $\mathbf{U}$ is the upper Cholesky factor of the inverse damped curvature. Here the label denotes the best candidate under this inverse-curvature diagonal proxy.

After the entire layer has been quantized and all inter-group compensation has been applied, we cast the reconstructed weights to BF16, as in the existing implementation, and recompute

$$L_{\text{final}} = \text{Tr}\left[(\mathbf{W}_0 - \hat{\mathbf{W}})\,\mathbf{H}_{\text{cal}}\,(\mathbf{W}_0 - \hat{\mathbf{W}})^\top\right], \tag{14}$$

where $\mathbf{W}_0$ is the original weight matrix and $\mathbf{H}_{\text{cal}}$ is the full, undamped cached curvature. Thus the evaluation retains cross-channel terms and measures the final reconstruction objective. Figure 5 normalizes this quantity by GPTQ with Min-Max scale selection at each layer.



(a) `up_proj` reconstruction loss (b) `down_proj` reconstruction loss

Figure 5: **Final reconstruction loss with groupwise GPTQ compensation.** Each panel has its own vertical axis and is normalized by the Min-Max baseline at the same layer. H-Scale lowers the final full-curvature loss in all 72 layer/projection pairs. "Best Scale (theory)" denotes the inverse-Cholesky candidate-scoring reference defined above.

H-Scale reduces the final reconstruction loss on every layer, and closely tracks Best Scale (theory). Weight-Guided Scale is weaker in both panels, so the Hessian-weighted local search remains effective after GPTQ compensation.

G Multimodal Quantization

Quantization and calibration. We apply NVFP4 to the language component of Qwen3.8-27B. The vision encoder and the protected gating, convolution, normalization, embedding, and output-head modules remain unquantized. Against the BF16 reference, we compare RTN, 4over6, GPTQ, GPTAQ, and ArcQuant, each in a scale-only (`identity`) variant and an H-Scale (`diagH`) variant. Both variants search the same window of 16 E4M3 scale codes, $\delta = -9, \ldots, 6$. The scale-only variant scores these candidates with uniform weights, and H-Scale scores them by calibration-input energy. Calibration statistics are estimated from a budget of $160 \times 4096 = 655{,}360$ tokens drawn from LLaVA-Instruct-150K and COCO train2017, which includes 1,280 images and 340,774 visual tokens (52.00% of the budget).

Evaluation protocol. Following the Qwen3.8 model card (Qwen Team, 2026), all evaluations use thinking mode at the low reasoning effort. We adopt the sampling hyperparameters recommended there for thinking mode—temperature 1.0, top-$p = 0.95$, top-$k = 20$, and presence penalty 0—so that generation follows the official recipe rather than a custom decoding setup. Text tasks are GPQA Diamond (Rein et al., 2024), IFBench (Pyatkin et al., 2025), and LiveCodeBench v5 (Jain et al., 2025). Multimodal tasks are RealWorldQA (xAI, 2024), ERQA (Gemini Robotics Team et al., 2025), MathVision (Wang et al., 2024), OCRBench (Liu et al., 2024), and ChartQA (Masry et al., 2022).

Table 13: Text and multimodal evaluation of Qwen3.8-27B. All quantized configurations use W4A4 NVFP4; the vision encoder remains in BF16. Avg. is the unweighted mean of the eight listed tasks. Bold marks the higher score within each quantization-method pair.

Method	Avg.	GPQA-D	IFBench	LCB-v5	RealWorldQA	ERQA	MathVision	OCRBench	ChartQA
BF16	75.40	83.69	65.77	64.94	81.70	61.50	86.32	85.80	73.44
RTN	73.31	80.10	64.13	61.22	81.01	59.75	84.67	**85.30**	70.32
+ H-Scale	**74.73**	**80.76**	**65.51**	**66.42**	**81.83**	**61.75**	**85.72**	84.60	**71.28**
4over6	73.26	81.46	64.98	58.53	**81.44**	59.50	84.38	**85.60**	70.20
+ H-Scale	**74.54**	**81.62**	**66.09**	**62.22**	81.31	**62.50**	**84.54**	84.50	**73.52**
GPTQ	74.11	81.21	64.87	**64.95**	81.31	**60.75**	84.51	84.60	70.64
+ H-Scale	**74.63**	**83.28**	**65.73**	64.94	**82.75**	59.00	**85.13**	**85.30**	**70.88**
GPTAQ	73.77	79.80	64.25	**63.98**	**81.31**	59.75	84.64	84.30	72.16
+ H-Scale	**74.18**	**81.21**	**65.26**	63.11	81.05	**60.25**	**85.33**	**84.90**	**72.32**
ArcQuant	73.75	81.26	63.25	62.62	81.05	**62.00**	84.54	84.80	70.48
+ H-Scale	**74.46**	81.26	**64.04**	**65.88**	81.05	61.75	**85.36**	84.80	**71.56**

H Implementation Pseudocode

After the empirical validation, we spell out the implementation used in the experiments. Algorithms 1–3 summarize the standalone scale search and its integration into representative NVFP4 pipelines. The diagonal weights $\mathbf{h}$ are computed as $\mathrm{diag}(\mathbf{X}^\top\mathbf{X})$ from calibration activations, or reused from the Hessian statistics already available in GPTQ-like baselines. H-Scale only refines the per-group scale and keeps the NVFP4 grouping, FP4 value grid, and FP8 scale representation unchanged. In Algorithms 2 and 3, the red lines mark the modules newly added on top of the original baseline procedure.

The search is fully parallelizable. Different row-group pairs and different candidates in $\mathcal{S}_{r,i}$ are independent. The pseudocode below makes that data-parallel structure explicit.

For GPTQ, H-Scale is inserted only at the local quantization step: the vanilla call $\mathbf{q} = Q(\mathbf{w})$ is replaced by a Hessian-weighted NVFP4 scale search, while the blockwise GPTQ error computation and propagation remain unchanged.

I Evaluation Protocol Details

Benchmark suites. For compact table headers, we abbreviate MMLU-Redux as MMLU-R and LiveCodeBench as LCBench. Qwen3-30A3-Thinking uses the five-task suite in Table 1. Qwen3-30A3-Instruct and Qwen3-4B-Instruct (Yang et al., 2025) use the eight-task suite in Tables 2 and 3. For LLaMA-3.1-8B-Instruct (Grattafiori et al., 2024), we report a compact seven-task suite and omit tasks where the BF16 baseline is near random or zero, since they do not provide meaningful quantization comparisons.

Decoding protocol. Generation uses temperature 0.7, top-p=0.8, top-k=20, repetition penalty 1.0, and presence penalty 1.5. LiveCodeBench uses beam size 10, and AIME uses beam size 16. Because stochastic decoding and benchmark sampling can introduce run-to-run fluctuation, we evaluate each method three times under this protocol and report the mean score. We do not attach statistical significance to small deltas or to cases where a quantized variant slightly exceeds BF16. We use the fixed protocol to keep paired comparisons controlled across methods.

J Limitations and Future Work

Our experiments focus on NVFP4 with $g = 16$, whose E2M1 values and E4M3 local scales define the search space. Transfer to other formats, group sizes remains open. Future work will adapt the candidate construction to other hardware formats and investigate joint refinement of weight and activation scales.

Algorithm 1 HScaleGroupQuantize: H-Scale for NVFP4 scale refinement

Require: Weight matrix $\mathbf{W}$, diagonal Hessian weights $\mathbf{h}$, initial effective scales $\mathbf{s}^{\text{init}}$, group size $g = 16$, upward budget $U = 6$, candidate budget $K = 16$, global scale s_{g}, E4M3 code count M

Ensure: Quantized weight matrix $\hat{\mathbf{W}}$

for all row-group pairs (r, i) **do**
  $\mathbf{w} \leftarrow \mathbf{w}_{r,i}$, $\mathbf{h}_i \leftarrow$ aligned slice of $\mathbf{h}$
  $s^{\text{init}} \leftarrow$ aligned baseline-provided effective NVFP4 scale for $\mathbf{w}$
  $\ell^{\text{init}} \leftarrow Q_{\text{fp8}}(s^{\text{init}}/s_{\text{g}})$, $c \leftarrow \iota(\ell^{\text{init}})$ ▷ ι: E4M3 code index, i.e. the stored scale byte
  $D \leftarrow K - 1 - U$ ▷ downward budget
  for $\delta = -D, \ldots, U$ **do**
    $\ell^{(\delta)} \leftarrow \lambda_{\text{clip}(c+\delta,\,1,\,M)}$ ▷ $\delta > 0$: larger scale; $\delta = 0$: ℓ^{init}; $\delta < 0$: smaller
  end for
  $\mathcal{L} \leftarrow \left\{ \ell^{(\delta)} \;\middle|\; -D \le \delta \le U \right\}$
  $\mathcal{S} \leftarrow \left\{ s_{\text{g}}\,\ell \;\middle|\; \ell \in \mathcal{L} \right\}$
  for all $\tilde{s} \in \mathcal{S}$ **do**
    $\hat{\mathbf{w}}(\tilde{s}) \leftarrow Q_{\text{fp4}}(\mathbf{w}/\tilde{s})\,\tilde{s}$
    $E(\tilde{s}) \leftarrow \|(\mathbf{w} - \hat{\mathbf{w}}(\tilde{s})) \odot \sqrt{\mathbf{h}_i}\|_2^2$
  end for
  $\tilde{s}^* \leftarrow \arg\min_{\tilde{s} \in \mathcal{S}} E(\tilde{s})$
  Store $\hat{\mathbf{w}}_{r,i} \leftarrow \hat{\mathbf{w}}(\tilde{s}^*)$
end for
return $\hat{\mathbf{W}}$

Algorithm 2 Combining 4over6 with H-Scale

Require: Row-group $\mathbf{w}_{r,i}$, Hessian weights $\mathbf{h}_i$, upward budget U, candidate budget K, global scale s_{g}, 4over6 anchors $\mathcal{C} = \{4, 6\}$

Ensure: Quantized row-group $\hat{\mathbf{w}}_{r,i}$

$m \leftarrow \|\mathbf{w}_{r,i}\|_\infty$
for all $c \in \mathcal{C}$ **do**
  $s_c \leftarrow m/c$
  $\ell_c \leftarrow Q_{\text{fp8}}(s_c/s_{\text{g}})$, $\tilde{s}_c \leftarrow s_{\text{g}}\ell_c$
  $\hat{\mathbf{w}}_c \leftarrow Q_{\text{fp4}}(\mathbf{w}_{r,i}/\tilde{s}_c)\,\tilde{s}_c$
  $L_{\text{4over6}}(c) \leftarrow \|\mathbf{w}_{r,i} - \hat{\mathbf{w}}_c\|_2^2$
end for
$c^{\text{init}} \leftarrow \arg\min_{c \in \mathcal{C}} L_{\text{4over6}}(c)$
$s^{\text{init}} \leftarrow \tilde{s}_{c^{\text{init}}}$
$\hat{\mathbf{w}}_{r,i} \leftarrow \text{HScaleGroupQuantize}(\mathbf{w}_{r,i}, \mathbf{h}_i, s^{\text{init}}, U, K, s_{\text{g}})$
return the refined group $\hat{\mathbf{w}}_{r,i}$

Algorithm 3 Combining GPTQ with H-Scale

Require: Weight matrix $\mathbf{W}$, GPTQ Hessian $\mathbf{H}$, diagonal proxy $\mathbf{h}$, block size B, upward budget U, candidate budget K

Ensure: Quantized weight matrix $\hat{\mathbf{W}}$

Form the damped inverse-Cholesky factor $\mathbf{H}^{-1/2}$ as in GPTQ
Compute the layer-level NVFP4 global scale s_{g}
Initialize $\hat{\mathbf{W}} \leftarrow \mathbf{0}$
for $i = 1, 1+B, \ldots, n$ **do**
    $\mathbf{W}_i \leftarrow \mathbf{W}_{:,i:i+B-1}$, $\mathbf{H}_i \leftarrow \mathbf{H}^{-1/2}_{i:i+B-1,i:i+B-1}$
    $\mathbf{h}_i \leftarrow \mathbf{h}_{i:i+B-1}$
    $\mathbf{s}_i^{\mathrm{init}} \leftarrow$ baseline-provided effective NVFP4 scales for $\mathbf{W}_i$
    $\hat{\mathbf{W}}_i \leftarrow \mathrm{HScaleGroupQuantize}(\mathbf{W}_i, \mathbf{h}_i, \mathbf{s}_i^{\mathrm{init}}, U, K, s_{\mathrm{g}})$
    for all local columns j in the block **do**
        $\mathbf{e}_j \leftarrow (\mathbf{W}_{i,j} - \hat{\mathbf{W}}_{i,j})/\mathbf{H}_{i,jj}$
    end for
    $\hat{\mathbf{W}}_{:,i:i+B-1} \leftarrow \hat{\mathbf{W}}_i$
    Update the remaining weights using the standard GPTQ error propagation:

$$\mathbf{W}_{:,i+B:n} \leftarrow \mathbf{W}_{:,i+B:n} - \mathbf{E}_i \mathbf{H}^{-1/2}_{i:i+B-1,i+B:n}.$$

end for
return $\hat{\mathbf{W}}$