# NADA: UNLOCKING FAST AND ACCURATE NVFP4 PTQ THROUGH DISTORTION-AWARE OPTIMIZATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

NVFP4 provides a wide effective dynamic range while retaining hardware-native FP4 computation on NVIDIA Blackwell GPUs, enabling high quantization fidelity with efficient low-precision inference. However, accurate NVFP4 post-training quantization (PTQ) remains challenging, particularly for activations. Simply suppressing outliers does not guarantee lower activation quantization error, largely because the nonuniform E2M1 codebook induces uneven distortion. We identify that NVFP4 activation reconstruction is governed by three interdependent and sequentially coupled decisions at the tensor, channel, and block levels. This paper proposes an NVFP4 Activation Distortion-Aware (NADA) optimization framework for fast and accurate NVFP4 PTQ. NADA consists of tensor-level Distortion-Aware Activation Channel Clustering (DACC), channel-level Bounded Iterative Top-$k$ Rescaling (BITR), and block-level Analytical Block-Scale Optimization (ABSO). DACC and BITR are calibrated offline to reorganize activation channels and selectively rescale sensitive channels, respectively, while ABSO performs a search-free analytical scale update for dynamic activation blocks at runtime. Across eight models ranging from 135M to 70B, NADA achieves state-of-the-art accuracy while preserving high end-to-end inference efficiency. Code is available at `https://anonymous.4open.science/r/NADA-5B92`.

## 1 INTRODUCTION

Large language models (LLMs) (Grattafiori et al., 2024; Yang et al., 2025) deliver strong performance across a wide range of tasks, but their growing model size and inference cost hinder efficient deployment. Quantization (Liu et al., 2024; Nagel et al., 2020) addresses this challenge by representing tensors at reduced precision, lowering storage, memory bandwidth, and computational cost. Among existing quantization paradigms, post-training quantization (PTQ) (Frantar et al., 2023; Lin et al., 2024) is particularly attractive for LLMs because it avoids costly retraining and requires only a small calibration set. The emergence of hardware-native FP4 computation further increases the practical value of low-bit PTQ. In particular, NVIDIA Blackwell GPUs natively support NVFP4, which combines a wide effective dynamic range with efficient FP4 computation (NVIDIA, 2026). However, preserving accuracy under NVFP4 PTQ remains challenging, especially for activations.

Activation quantization is difficult due to varying and input-dependent distributions. Existing PTQ methods mainly mitigate this issue through outlier suppression by rescaling and rotations (Xiao et al., 2023; Ashkboos et al., 2024; Liu et al., 2025), while recent NVFP4 methods explore block-scale optimization (Cook et al., 2026; Lin & Wan, 2026). However, reducing the activation range does not necessarily reduce NVFP4 quantization error. Unlike uniform integer quantization, NVFP4 uses a non-uniform E2M1 codebook, causing blocks with similar ranges to exhibit different reconstruction errors depending on how their normalized values align with the code points. Accurate NVFP4 PTQ therefore requires distortion-aware optimization beyond range reduction alone.

We identify three interdependent and sequentially coupled decisions that govern NVFP4 activation reconstruction. At the tensor level, channel organization determines which activation values are clustered into each NVFP4 block at a given token, thereby shaping the within-block distributions (Cheng et al., 2026). At the channel level, equivalent activation-weight rescaling redistributes quantization difficulty between the two GEMM operands (Lin et al., 2024; Xiao et al., 2023). At the block level, the quantization scale determines how each dynamic activation block is mapped to the



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.

E2M1 codebook (Cook et al., 2026; Lin & Wan, 2026). These decisions are sequentially coupled, as channel organization shapes the distributions for channel-level rescaling, which in turn determines the block statistics for scale optimization. This motivates a coarse-to-fine optimization from tensor-level organization to channel-level refinement and block-level scale optimization.

In this paper, we propose an **N**VFP4 **A**ctivation **D**istortion-**A**ware (**NADA**) framework for fast and accurate NVFP4 PTQ. Our contributions are summarized as follows:

- We show that activation range reduction alone is insufficient for accurate NVFP4 quantization and identify the non-uniform distortion induced by the E2M1 codebook as an important source of reconstruction error.
- We formulate NVFP4 PTQ as a coarse-to-fine optimization problem with three interdependent components: tensor-level Distortion-Aware Activation Channel Clustering (DACC) for channel reorganization, channel-level Bounded Iterative Top-$k$ Rescaling (BITR) for sensitive-channel rescaling, and block-level Analytical Block-Scale Optimization (ABSO) for dynamic block-scale optimization.
- Extensive experiments across eight models ranging from 135M to 70B parameters show that NADA jointly optimizes these three levels through offline calibration and lightweight runtime refinement, achieving state-of-the-art NVFP4 PTQ accuracy while maintaining high inference efficiency.

## 2 Background and Related Work

### 2.1 LLM PTQ

Quantization has become a key technique for efficient LLM inference by representing tensors at reduced precision, thereby reducing storage requirements, memory bandwidth, and computational cost. Its practical importance has grown with the advent of hardware-native low-precision inference, exemplified by NVFP4 support in NVIDIA Blackwell GPUs (NVIDIA, 2025). Existing quantization methods generally fall into two paradigms: quantization-aware training (QAT) and PTQ. While QAT generally achieves higher accuracy, it incurs substantial retraining overhead. In contrast, PTQ requires only a limited calibration set and avoids full retraining, making it a more practical choice.

For LLMs, PTQ can target weights, activations, or the KV cache. In this work, we focus on NVFP4 W4A4 quantization for linear layers. Existing PTQ methods for linear layers can be broadly divided into weight-only and weight-activation quantization. Weight-only PTQ quantizes static weights while preserving activations at full precision. Representative methods include GPTQ (Frantar et al., 2023), which performs Hessian-guided adaptive weight quantization with error compensation, and AWQ (Lin et al., 2024), which applies equivalent rescaling to protect salient weight channels. Weight-activation PTQ further quantizes both operands of general matrix multiplication (GEMM), enabling low-precision compute for the entire linear operation. Representative approaches in this category include SmoothQuant (Xiao et al., 2023), QuaRot (Ashkboos et al., 2024), and SpinQuant (Liu et al., 2025). Existing PTQ techniques can be further categorized into three levels according to the granularity at which optimization objectives are defined and coupled:

- **Tensor-Level**: Methods at this level transform or reorganize the full hidden dimension. Rotation-based approaches, including QuaRot (Ashkboos et al., 2024), SpinQuant (Liu et al., 2025), OSTQuant (Hu et al., 2025), and MR-GPTQ (Egiazarian et al., 2026), apply global or block-wise rotations to reshape tensor distributions, while PermuQuant (Cheng et al., 2026) and ARCQuant (Meng et al., 2026) reorganize channel layouts to reduce quantization error.
- **Channel-Level**: Channel-level methods optimize channel-wise transformations, typically through activation and weight rescaling. SmoothQuant (Xiao et al., 2023) balances their dynamic ranges, AWQ (Lin et al., 2024) searches for scales that protect salient weight channels, and OmniQuant (Shao et al., 2024) learns equivalent transformations using reconstruction objectives.
- **Block-Level**: Block-level methods, also referred to as group-level methods, optimize quantization parameters shared by a small group of values. For static weights, block-wise scales

can be optimized offline using MSE- or reconstruction-based objectives (NVIDIA, 2026; Egiazarian et al., 2026; Bao et al., 2026). For input-dependent activations, however, block scales must be determined efficiently at runtime, making their optimization crucial to end-to-end inference performance.

### 2.2 NVFP4 PTQ

Unlike conventional linear quantization, which maps values to uniformly spaced integer levels using a scale and optionally a zero point, FP4 uses a non-uniform floating-point codebook whose spacing increases with magnitude. This representation offers a wider relative dynamic range and is well suited to hardware-native low-precision computation, but its irregular code-point spacing makes accuracy depend on both the value range and codebook alignment. GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2024), and SmoothQuant (Xiao et al., 2023) primarily target conventional linear quantization, whereas recent methods such as MR-GPTQ (Egiazarian et al., 2026), Four-Over-Six (Cook et al., 2026), and ARCQuant (Meng et al., 2026) specifically optimize microscaling FP4 formats.

NVFP4 and MXFP4 are two representative hardware-native FP4 formats. Compared with the 32-element blocks and E8M0 power-of-two scales of MXFP4, NVFP4 uses 16-element blocks and finer E4M3 scales, enabling more accurate adaptation to local value distributions (NVIDIA, 2025). We therefore focus on NVFP4; a detailed format comparison is provided in Appendix A. Specifically, NVFP4 represents a tensor using one FP32 tensor scale $s_T$, one E4M3 block scale $s_{B(i)}$ per 16 elements, and the E2M1 codebook. An element $x_i$ in block $B(i)$ is reconstructed as:

$$\hat{x}_i = s_T s_{B(i)} q_i, \qquad q_i \in \mathcal{C}_{\mathrm{E2M1}}, \qquad \mathcal{C}_{\mathrm{E2M1}} = \{\pm 0, \pm 0.5, \pm 1, \pm 1.5, \pm 2, \pm 3, \pm 4, \pm 6\}.$$

### 2.3 Challenges in NVFP4 Weight-Activation Quantization

Prior work has shown that activations are generally more difficult to quantize than weights in LLMs, and thus demand more careful handling (Xiao et al., 2023). This difficulty stems from two aspects:

- **Accuracy.** Unlike static weights, activations exhibit wider dynamic ranges that vary with the input and contain more pronounced channel- and token-wise outliers. In NVFP4, every 16 values share a single block scale; consequently, a single large outlier can dominate scale selection and degrade the representation fidelity of the remaining elements. As a result, low-bit activation quantization often incurs substantially greater accuracy loss than its weight-only counterpart.
- **Inference efficiency.** Weights and their associated quantization parameters can be calibrated and stored offline, whereas activations are generated dynamically and must be quantized on the inference critical path. More sophisticated activation-side optimization therefore directly increases runtime overhead. For instance, block-scale selection methods such as Four-Over-Six and ScaleSweep evaluate multiple candidates based on quantization error (Cook et al., 2026; Lin & Wan, 2026). While effective, such online search introduces additional quantization and comparison operations at inference time. Consequently, an effective NVFP4 W4A4 method must improve activation quantization accuracy without incurring prohibitive online computation.

## 3 Motivation

As discussed, activation quantization is affected not only by large dynamic ranges but also by the input-dependent distributions of dynamic activation blocks. Existing methods mainly suppress outliers, balance channel ranges, or search for improved block scales. To understand why these operations do not always produce lower NVFP4 quantization error, we examine the quantization process from the perspective of individual values and their positions relative to the E2M1 codebook.

**From quantization error to codebook distortion.** Consider an element $x_i$ belonging to block $b$. Let $s_T$ denote the tensor scale and $s_{B,b}$ denote the E4M3 scale of block $b$. The normalized value is:

$$z_i = \frac{x_i}{s_T s_{B,b}}, \qquad i \in b.$$

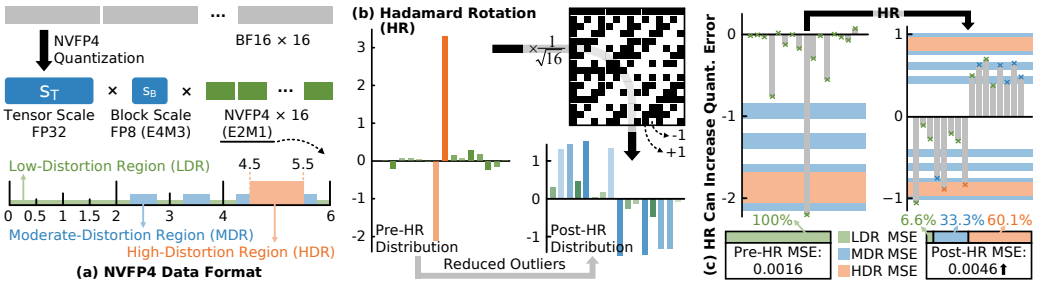


Figure 1: (a) The non-uniform E2M1 codebook induces low-, moderate-, and high-distortion regions within the NVFP4 representable range. (b) Hadamard rotation reduces activation outliers and narrows the distribution. (c) Despite the reduced dynamic range, rotation can move normalized values into higher-distortion regions and consequently increase activation quantization error.

Using the E2M1 codebook defined in Section 2, the reconstructed value is:

$$q_i = Q_{\mathrm{E2M1}}(z_i) = \arg\min_{q \in \mathcal{C}_{\mathrm{E2M1}}} |z_i - q|, \qquad \hat{x}_i = s_T s_{B,b} q_i.$$

We define the normalized distortion magnitude and its squared form as:

$$\delta(z_i) = |z_i - q_i|, \qquad d(z_i) = \delta(z_i)^2.$$

The resulting element-wise quantization error is:

$$e_i = (x_i - \hat{x}_i)^2 = (s_T s_{B,b})^2 d(z_i),$$

where $e_i$ denotes the squared reconstruction error of activation element $x_i$ in block $b$. This relation shows that the element-wise quantization error is jointly determined by the effective scale $s_T s_{B,b}$ and the position of the normalized value $z_i$ relative to the E2M1 code points.

**Distortion regions.** Because the E2M1 code points become increasingly sparse away from zero, the nearest-codepoint distance is non-uniform over the representable range. For $u = |z| \in [0, 6]$, we define the three regions: high-distortion region (HDR), moderate-distortion region (MDR), and low-distortion region (LDR). These thresholds follow directly from the three positive E2M1 code-point gaps, 0.5, 1, and 2, whose maximum nearest-codepoint distances are 0.25, 0.5, and 1, respectively.

$$\begin{aligned} \mathrm{HDR} &= (4.5, 5.5), && 0.5 < \delta(u) \le 1, \\ \mathrm{MDR} &= (2.25, 2.75) \cup (3.25, 3.75) \cup (4.25, 4.5] \cup [5.5, 5.75), && 0.25 < \delta(u) \le 0.5, \\ \mathrm{LDR} &= [0, 6] \setminus (\mathrm{MDR} \cup \mathrm{HDR}), && 0 \le \delta(u) \le 0.25. \end{aligned}$$

The same partition applies symmetrically to negative values. Values outside $[-6, 6]$ are clipped to the boundary code points and are treated separately as overflow values.

**Limitations of maximum-based scaling.** The standard NVFP4 AbsMax rule determines the block scale from the largest magnitude and maps that maximum close to the boundary code point 6. Although this protects the block maximum, the scale calculation does not explicitly consider the remaining elements. A block can therefore represent its maximum accurately while still accumulating substantial distortion from values located in the moderate- and high-distortion regions.

**Outlier reduction is not sufficient.** Rotation-based transformations reduce outliers by mixing channels and compressing the overall activation range. However, they optimize the range rather than the distortion regions induced by the non-uniform E2M1 code points. As shown in Figure 1(b-c), Hadamard rotation reduces the largest activation magnitude but moves several normalized values from the LDR into the MDR and HDR. The resulting activation MSE consequently increases despite the reduced dynamic range. Therefore, a narrower distribution does not necessarily imply lower NVFP4 quantization error.

**Activation-side evidence.** To identify whether this behavior originates from weight or activation quantization, we separately evaluate weight-only and activation-only quantization. Compared with

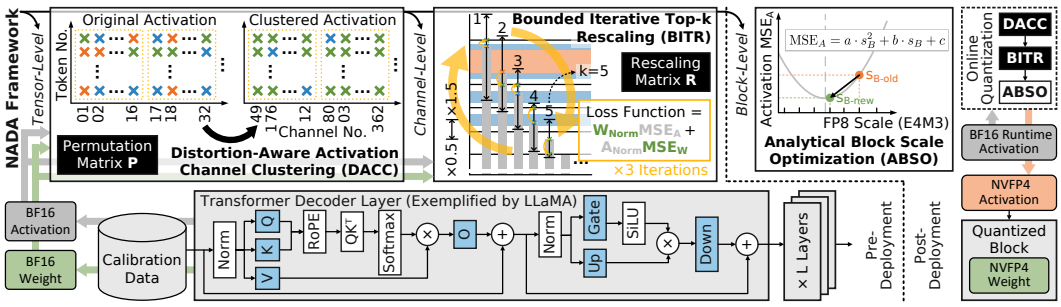


Figure 2: Overview of NADA. DACC, BITR, and ABSO progressively optimize NVFP4 activation quantization at the tensor, channel, and block levels. The lower panel shows their placement in a LLaMA-style decoder and distinguishes offline calibration and fusion from online processing.

GPTQ, MR-GPTQ improves the weight-only result, indicating that its micro-rotation and GPTQ compensation remain effective for weight reconstruction. In contrast, its activation-only result becomes worse, showing that the transformed activations can be less compatible with the E2M1 codebook. This observation identifies activation distortion, rather than weight, as a major source of the observed degradation. Appendix B details weight-only and activation-only comparisons.

**Design objective.** The preceding analysis identifies three coupled decisions that affect NVFP4 reconstruction: the tensor-level decision of which channels share a block, the channel-level decision of how their magnitudes are rescaled, and the block-level decision of selecting the optimal scale for each runtime block. These decisions are interdependent and sequentially coupled, as each stage operates on the representation produced by the preceding one. NADA therefore first permutes channels within each tensor using DACC, then refines channel magnitudes with BITR, and finally adapts runtime block scales with ABSO.

# 4 Methodology

## 4.1 NADA Framework

Figure 2 presents NADA, a three-step distortion-aware optimization pipeline for NVFP4 W4A4 PTQ. Rather than optimizing the activation range alone, NADA progressively reduces the mismatch between activation values and the non-uniform NVFP4 representation. DACC first reorganizes activation blocks at the tensor level, BITR subsequently refines sensitive channels under the resulting layout, and ABSO finally updates the scale of each dynamic activation block. This order moves from coarse tensor-level organization to fine block-scale refinement.

NADA separates offline optimization from online execution, as shown in Figure 2. DACC and BITR are determined once using calibration data, after which their corresponding weight-side transformations are applied offline and compatible activation-side operations can be fused into adjacent operators when permitted by the computation graph. In contrast, ABSO operates online because activation blocks are input-dependent and their scales must be determined dynamically. Its single-step update can be integrated directly into activation quantization with minimal additional overhead.

## 4.2 DACC: Distortion-Aware Activation Channel Clustering

NVFP4 assigns an independent E4M3 scale to every 16-element activation block. Consequently, changing which channels form a block changes its block maximum and effective scale, potentially moving its normalized values into lower-distortion regions. DACC exploits this flexibility by constructing a channel permutation from calibration activations.

Let $X \in \mathbb{R}^{T \times C}$ contain $T$ calibration tokens and $C$ input channels, and let $B = 16$ denote the NVFP4 block size. Here, $c$ denotes a channel index and $W_{:,c}$ denotes the corresponding column of the linear weight matrix. DACC partitions the channels into $K = C/B$ clusters, each of which forms

one NVFP4 activation block at every token. It therefore permutes complete channels consistently across all tokens rather than exchanging individual activation values between blocks.

**Pairwise proxy distortion.** The exact block scale is unavailable before all $B$ channels have been assigned. DACC therefore estimates the compatibility of two channels using the procedure described in Appendix C. At each token, the larger magnitude is mapped to the maximum E2M1 value of 6 to determine a proxy effective scale, under which the squared quantization distortion of the smaller magnitude is computed and accumulated. This is called *proxy distortion*, rather than exact quantization error, because the other channels and the final E4M3 block-scale projection are not yet known.

**Greedy cluster construction.** DACC constructs the permutation in three stages. First, it selects the channel with the largest weight-aware importance score as the first seed. This channel is likely to determine the block maximum across many tokens, making the subsequent proxy scale more stable. Second, the remaining cluster seeds are selected by maximizing their cumulative proxy distortion from the existing seeds, separating channels that are likely to cause severe distortion when placed together. Third, the remaining channels are processed from high to low importance and assigned to the compatible non-full cluster with the smallest proxy distortion.

The weight-aware importance of channel $c$ is defined as its accumulated activation magnitude multiplied by $\|W_{:,c}\|_2^{\alpha}$, approximating the influence of that channel on the linear-layer output. We set $\alpha = 2$, which achieves the best overall performance in our empirical study; a detailed sensitivity analysis is provided in Appendix C.2. This importance score is used only to determine the cluster seeds and channel processing order. During clustering, each $\mathbf{m}_j$ represents the token-wise maximum envelope of cluster $\mathcal{G}_j$, rather than an individual channel. DACC greedily assigns channels according to activation proxy distortion, while maintaining cumulative seed distances and cluster maximum envelopes. The resulting channel permutation is constructed once during calibration without iterative reassignment. The complete procedure is provided in Appendix C.1.

Appendix E further analyzes the mechanism of DACC by comparing it with five independent random permutations and four deterministic ordering controls. DACC achieves the lowest activation MSE among all permutation-only methods and also minimizes the absolute MSE contributed by MDR/HDR values. Under the resulting DACC layout, BITR further reduces the joint weight-activation reconstruction error (Table 8).

### 4.3 BITR: Bounded Iterative Top-$k$ Rescaling

DACC reorganizes activation blocks at the tensor level but does not explicitly refine individual channel magnitudes. Under the resulting layout, BITR further adjusts the most influential channels to reduce the remaining joint weight-activation reconstruction error. Specifically, it rescales activation channels and inversely rescales the corresponding weight columns, thereby preserving the full-precision linear operation.

The AWQ-style method used by Four-Over-Six (Cook et al., 2026) selects channel scales according to output reconstruction MSE. Although accurate, every candidate evaluation requires quantizing the full weight and activation tensors and recomputing the layer output. We empirically find that multiple rounds can further improve accuracy by updating block scales and code assignments, but at a substantial calibration cost (Appendix D).

BITR reduces this cost through selective search and a lightweight objective. Within each 16-channel block, it optimizes only the top-$k$ channels with the largest product of activation and weight squared norms, which estimates their influence on the layer output. Instead of repeatedly computing output MSE, BITR evaluates each candidate rescaling $r$ using

$$\mathcal{L}_{\mathrm{BITR}}(r) = W_{\mathrm{Norm}}\,\mathrm{MSE}_A(r) + A_{\mathrm{Norm}}\,\mathrm{MSE}_W(r),$$

where $W_{\mathrm{Norm}} = (\mathrm{MSE}_A(1) + \epsilon)^{-1}$ and $A_{\mathrm{Norm}} = \lambda(\mathrm{MSE}_W(1) + \epsilon)^{-1}$, with $r = 1$ denoting identity rescaling. We use $\lambda = 1$ in the GPTQ-based configuration. This normalized operand-level objective avoids repeatedly reconstructing the complete layer output.

Benefiting from the block layout produced by DACC, BITR restricts its search to small local adjustments. We optimize the top five channels in each block, evaluate 21 candidates uniformly distributed between $0.5$ and $1.5$, and perform three coordinate-update rounds. Block scales and E2M1 assignments are recomputed during candidate evaluation. On Qwen3-8B, 3-round BITR requires 31.2

minutes, compared with 68.3 minutes for 3-round full-channel AWQ-style search, while achieving a similar average perplexity (16.073 versus 16.067). See details in Appendix D.

**Equivalent transformation.** Let $P$ denote the permutation produced by DACC and $R$ the diagonal rescaling produced by BITR. Their joint transformation preserves the full-precision linear operation:

$$RR^{-\top} = I, \quad PP^{\top} = I, \quad (XPR)(WPR^{-1})^{\top} = XW^{\top}.$$

The transformed weight $WPR^{-1}$ is materialized before weight quantization. The placement and fusion of activation-side transformations are detailed in Appendix F.

### 4.4 ABSO: Analytical Block-Scale Optimization

After DACC reorganizes the tensor layout and BITR refines sensitive channel magnitudes, the remaining optimization lies at the block level, where each dynamic activation block requires an appropriate scale. Exhaustive candidate-scale search can improve reconstruction accuracy but incurs non-negligible runtime overhead. We therefore propose ABSO, whose key idea is to replace runtime candidate-scale search with a closed-form least-squares update for each activation block.

ABSO first initializes the block scale using AbsMax and quantizes each activation value to an E2M1 code point. With these code assignments fixed, the optimal block scale can be obtained analytically by minimizing the reconstruction error. Let $x_i$ denote the activation value and $q_i^{(0)}$ its initial signed E2M1 assignment. After factoring out the fixed tensor scale $s_T$, the block scale is updated as:

$$s_B^{\star} = \arg\min_{s_B} \sum_{i \in b} \left( \frac{x_i}{s_T} - s_B q_i^{(0)} \right)^2 = \frac{\sum_{i \in b} \left( \frac{x_i}{s_T} \right) q_i^{(0)}}{\sum_{i \in b} \left( q_i^{(0)} \right)^2}.$$

The continuous solution $s_B^{\star}$ is then projected to a valid E4M3 block scale, and the E2M1 assignments are recomputed using the updated scale. The closed-form solution is exact for the least-squares objective under fixed code assignments, while the subsequent E4M3 projection and E2M1 reassignment map this solution back to the discrete NVFP4 representation. For an all-zero block, ABSO retains the initial scale.

Because ABSO optimizes the scale under fixed initial assignments rather than exhaustively evaluating the discrete NVFP4 objective, it does not guarantee the globally optimal scale. Instead, it trades exhaustive discrete optimality for substantially lower runtime cost. Each input-dependent activation block requires only block-wise reductions and one division, without enumerating or evaluating multiple candidate scales. Empirically, after DACC and BITR preprocessing, a single ABSO update consistently improves quantization accuracy, while additional updates provide negligible benefit and may slightly increase perplexity. We therefore adopt a one-step update by default, with the corresponding comparison reported in Appendix H.

## 5 Experiments

### 5.1 Experimental Setup

**Models and baselines.** We evaluate NADA on eight decoder-only language models from three families: SmolLM2-135M (Allal et al., 2025); Qwen3-0.6B, Qwen3-1.7B, Qwen3-8B, and Qwen3-32B (Yang et al., 2025); and Llama3.2-1B, Llama3-8B, and Llama3-70B (Grattafiori et al., 2024). We compare against the BF16 reference and five NVFP4 PTQ baselines: RTN, GPTQ (Frantar et al., 2023), MR-GPTQ (Egiazarian et al., 2026), Four-Over-Six with AWQ-style rescaling (Cook et al., 2026), and ARCQuant (Meng et al., 2026).

**Quantization settings.** All quantized methods target NVFP4 W4A4 for the linear layers in the Transformer blocks. We use 128 randomly sampled C4 calibration sequences with a sequence length of 2,048 and fix the calibration seed to zero. Unless otherwise specified, NADA uses the hyperparameters described in Sections 4.2-4.4. All methods use identical calibration data, quantized-layer coverage, and evaluation scripts.

Table 1: Progressive ablation of the NADA pipeline under NVFP4 W4A4 quantization. All NADA variants use GPTQ as the weight-quantization backbone. Lower perplexity is better.

| Method | DACC | BITR | ABSO | SmolLM2-135M | | Qwen3-1.7B | | Llama3-8B | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | WKT2↓ | C4↓ | WKT2↓ | C4↓ | WKT2↓ | C4↓ |
| BF16 | ✗ | ✗ | ✗ | 15.444 | 21.355 | 16.719 | 22.394 | 6.137 | 9.446 |
| GPTQ | ✗ | ✗ | ✗ | 21.118 | 28.148 | 17.968 | 23.898 | 6.774 | 10.520 |
| NADA variants | ✓ | ✗ | ✗ | 20.663 | 27.567 | 17.252 | 23.552 | 6.752 | 10.527 |
| | ✓ | ✓ | ✗ | 18.908 | 25.482 | 17.361 | 23.366 | 6.739 | 10.524 |
| | ✓ | ✓ | ✓ | **18.841** | **25.266** | **17.120** | **23.208** | **6.711** | **10.441** |

**Evaluation.** Perplexity is evaluated on WikiText-2 (WKT2) (Merity et al., 2016) and C4 (Raffel et al., 2020) test sets using non-overlapping sequences of 2,048 tokens. We additionally evaluate ten tasks with the LM Evaluation Harness (Biderman et al., 2026): PIQA (Bisk et al., 2020), WinoGrande (Sakaguchi et al., 2021), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), MMLU (Hendrycks et al., 2021), ARC-Easy, ARC-Challenge (Clark et al., 2018), LAMBADA (Paperno et al., 2016), GSM8K (Cobbe et al., 2021), and MBPP (Austin et al., 2021). Detailed task configurations are listed in Appendix I.2.

### 5.2 Ablation and Analysis

We perform a progressive ablation of the NADA pipeline on three representative models spanning different model families and parameter scales. Starting from the GPTQ baseline, DACC, BITR, and ABSO are added sequentially following the coarse-to-fine order introduced in Section 4.1. As shown in Table 1, the full pipeline achieves the best overall performance. Although the individual gains are not strictly uniform across all configurations, the three components achieve the strongest overall performance when combined, highlighting their complementary and synergistic effects across tensor, channel, and block levels. The contribution of each component varies across model scales. DACC provides clear improvements on SmolLM2-135M and Qwen3-1.7B, while BITR yields the largest additional gains on SmolLM2-135M. ABSO further improves all six model-dataset settings over DACC+BITR, confirming the consistent benefit of runtime block-scale refinement. Appendices D and H further show that BITR approaches multi-round full-channel search at substantially lower calibration cost, while a single ABSO update is sufficient after DACC and BITR preprocessing. Overall, the results support the complementary roles of the three components in the coarse-to-fine NADA pipeline.

### 5.3 Main Results

Table 2 compares NADA with representative NVFP4 W4A4 PTQ baselines across four models ranging from 1B to 8B parameters. NADA consistently achieves the lowest perplexity on both WikiText-2 and C4 and the highest downstream recovery among all quantized methods. Detailed per-task results are reported in Table 11 in Appendix I.1. The gains are particularly pronounced on the smaller Llama3.2-1B and Qwen3-1.7B models, where NVFP4 quantization causes larger performance degradation. On Qwen3-1.7B, for example, NADA improves downstream recovery by 2.64, 1.91, 2.86, and 3.75 percentage points over GPTQ, MR-GPTQ, Four-Over-Six, and ARCQuant, respectively. For the larger 8B models, where quantization degradation is already relatively small, NADA still provides consistent improvements and reaches 98.08% recovery on Qwen3-8B. Overall, these results demonstrate the robustness of NADA across different model families and scales.

Results on additional smaller models (135M and 0.6B) and larger models (32B and 70B) are reported in Appendix J. NADA maintains competitive performance across this broader range of model scales.

### 5.4 Efficiency and Deployment

We evaluate the deployment efficiency of NADA on Qwen3-8B using a single NVIDIA RTX 5090D GPU under two complementary settings. The first uses vLLM with native packed NVFP4 operators to assess serving-oriented performance. The second uses Hugging Face Transformers (HF) to

Table 2: NADA main results. **Rec.** denotes the ratio between the macro-average score over the ten downstream tasks and the corresponding BF16 macro-average. Each task score is computed using its standard evaluation metric.

| Method | Llama3.2-1B | | | Qwen3-1.7B | | | Llama3-8B | | | Qwen3-8B | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | WKT2↓ | C4↓ | Rec.↑ | WKT2↓ | C4↓ | Rec.↑ | WKT2↓ | C4↓ | Rec.↑ | WKT2↓ | C4↓ | Rec.↑ |
| BF16 | 9.750 | 14.016 | 100.00 | 16.719 | 22.394 | 100.00 | 6.137 | 9.446 | 100.00 | 11.994 | 19.197 | 100.00 |
| RTN | 11.523 | 16.923 | 90.47 | 18.546 | 24.166 | 90.21 | 6.869 | 10.683 | 95.26 | 12.502 | 19.836 | 97.47 |
| GPTQ | 11.068 | 16.183 | 92.62 | 17.968 | 23.898 | 92.13 | 6.774 | 10.520 | 95.85 | 12.495 | 19.992 | 97.35 |
| MR-GPTQ | 11.223 | 16.400 | 91.92 | 17.652 | 23.459 | 92.86 | 6.864 | 10.629 | 95.56 | 12.704 | 20.037 | 97.28 |
| 4/6 + AWQ | 11.373 | 16.803 | 91.16 | 18.461 | 24.724 | 91.91 | 6.747 | 10.499 | 94.82 | 12.403 | 19.818 | 97.04 |
| ARCQuant | 11.364 | 16.805 | 91.06 | 18.840 | 23.702 | 91.02 | 6.822 | 10.663 | 94.62 | 12.867 | 20.306 | 96.63 |
| **NADA** | **10.996** | **15.985** | **94.40** | **17.120** | **23.208** | **94.77** | **6.711** | **10.441** | **95.92** | **12.396** | **19.751** | **98.08** |

Table 3: End-to-end comparison with the official Four-Over-Six implementation on Qwen3-8B. GiB denotes peak GPU memory in gibibytes, and tok/s denotes throughput in tokens per second.

| Method | BS=1 | | BS=2 | | BS=4 | | BS=8 | | BS=16 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | GiB ↓ | tok/s ↑ | GiB ↓ | tok/s ↑ | GiB ↓ | tok/s ↑ | GiB ↓ | tok/s ↑ | GiB ↓ | tok/s ↑ |
| 4/6 | 7.75 | 14.32 | 7.82 | 27.94 | 8.03 | 57.11 | 8.50 | 113.39 | 9.73 | 222.32 |
| NADA | 7.75 | **17.90** | 7.82 | **35.96** | 8.03 | **71.34** | 8.50 | **140.75** | 9.72 | **271.17** |
| Ratio | 1.00× | **1.25×** | 1.00× | **1.29×** | 1.00× | **1.25×** | 1.00× | **1.24×** | 1.00× | **1.22×** |

enable a controlled comparison with the official Four-Over-Six CUDA implementation. In both settings, each request contains 512 input tokens and generates exactly 128 output tokens with greedy decoding. We perform two warm-up runs followed by five measured runs and report the median output-token throughput and generation-stage peak GPU memory. Both vLLM and HF deployments use the same fusion strategy. The weight-side DACC and BITR transformations are folded offline, while the remaining activation-side permutation and rescaling are fused with ABSO, E4M3 scale conversion, and E2M1 packing in the activation-quantization kernels. This avoids materializing intermediate transformed activations and reduces additional memory traffic.

Under vLLM, RTN, GPTQ, and MR-GPTQ use offline optimization, while NADA additionally applies search-free ABSO for online activation refinement. As shown in Appendix G, NADA outperforms BF16 across all evaluated batch sizes, while remaining within 12.2% of the fastest quantized baseline in throughput and adding at most 0.21 GiB of peak memory. This demonstrates that the accuracy gains in Table 2 are achieved with modest deployment overhead.

We further compare NADA with the official Four-Over-Six implementation using the same Transformers generation loop, the same 252 quantized linear layers, and the same SM120 CUTLASS NVFP4 GEMM. Four-Over-Six performs online MSE-based candidate-scale selection, whereas NADA uses search-free ABSO. As shown in Table 3, NADA achieves 1.22×-1.29× the end-to-end throughput of Four-Over-Six, demonstrating the efficiency of search-free activation refinement.

## 6 Conclusion

We present NADA, a distortion-aware NVFP4 PTQ framework motivated by the observation that range reduction alone does not guarantee lower quantization error. NADA coordinates tensor-level DACC, channel-level BITR, and block-level ABSO, combining offline calibration with search-free online optimization. Extensive experiments demonstrate competitive accuracy, including the best quantized perplexity and average downstream-task recovery. Fused deployment maintains practical vLLM throughput and achieves 1.22×-1.29× the throughput of official Four-Over-Six. By providing an accurate and efficient NVFP4 PTQ solution, NADA paves the way for practical W4A4 deployment of LLMs on modern FP4-accelerated hardware. Future work will adapt NADA to other FP4 formats and extend NADA to mixture-of-experts and multimodal models.

### AI USE STATEMENT

AI-assisted tools were used to support code debugging, LaTeX formatting, table organization, and language polishing. All methodological decisions, mathematical derivations, citations, code changes, and experimental results were independently checked and verified by the authors. The authors remain fully responsible for the accuracy, originality, and final content of this paper.

### REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide the NADA implementation through the repository linked in the abstract. Our experiments use publicly available pretrained models and benchmark datasets. The quantization and calibration settings are described in Section 5.1, while the method hyperparameters are specified in Sections 4.2, 4.3, and 4.4. Additional algorithmic details are provided in Appendix C, and task-specific metrics, few-shot settings, and evaluation subsets are documented in Appendix I.2. Hardware configurations and inference benchmarking procedures are described in Section 5.4 and Appendix G. We further report sensitivity to calibration data and random seeds in Appendices K and M.

## REFERENCES

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. SmolLM2: When Smol Goes Big – Data-Centric Training of a Small Language Model, 2025. URL https://arxiv.org/abs/2502.02737.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. QuaRot: Outlier-Free 4-Bit Inference in Rotated LLMs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 100213–100240, 2024. doi: 10.52202/079017-3180. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/b5b939436789f76f08b9d0da5e81af7c-Paper-Conference.pdf.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program Synthesis with Large Language Models, 2021. URL https://arxiv.org/abs/2108.07732.

Chengzhu Bao, Xianglong Yan, Zhiteng Li, Guangshuo Qin, Guanghua Yu, and Yulun Zhang. SOAR: Scale Optimization for Accurate Reconstruction in NVFP4 Quantization, 2026. URL https://arxiv.org/abs/2605.12245.

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, Anthony DiPofi, Julen Etxaniz, Benjamin Fattori, Jessica Zosa Forde, Charles Foster, Jeffrey Hsu, Mimansa Jaiswal, Wilson Y. Lee, Haonan Li, Charles Lovering, Niklas Muennighoff, Ellie Pavlick, Jason Phang, Aviya Skowron, Samson Tan, Xiangru Tang, Kevin A. Wang, Genta Indra Winata, François Yvon, and Andy Zou. Lessons from the Trenches on Reproducible Evaluation of Language Models, 2026. URL https://arxiv.org/abs/2405.14782.

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. URL https://ojs.aaai.org/index.php/AAAI/article/view/6239.

Yongsen Cheng, Kai Liu, Kaiwen Tao, Junxian Li, Zhixin Wang, Zhikai Chen, Renjing Pei, and Yulun Zhang. PermuQuant: Lowering Per-Group Quantization Error by Reordering Channels for Diffusion Models, 2026. URL https://arxiv.org/abs/2605.09503.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, June 2019. doi: 10.18653/v1/N19-1300. URL https://aclanthology.org/N19-1300/.

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, 2018. URL https://arxiv.org/abs/1803.05457.

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, 2026. URL https://arxiv.org/abs/2512.02010.

Vage Egiazarian, Roberto 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. In *International Conference on Learning Representations*, volume 2026, pp. 113529–113563, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/b87bb4f6346d727b265088235e5bc389-Paper-Conference.pdf.

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. arXiv:2210.17323.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan

Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov,

Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The Llama 3 Herd of Models, 2024. URL https://arxiv.org/abs/2407.21783.

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*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Xing Hu, Yuan Cheng, Dawei Yang, Zukang Xu, Zhihang Yuan, Jiangyong Yu, Chen Xu, Zhe Jiang, and Sifan Zhou. OstQuant: Refining Large Language Model Quantization with Orthogonal and Scaling Transformations for Better Distribution Fitting, 2025. URL https://arxiv.org/abs/2501.13987.

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 On-Device LLM Compression and Acceleration. In *Proceedings of Machine Learning and Systems*, volume 6, pp. 87–100, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/file/42a452cbafa9dd64e9ba4aa95cc1ef21-Paper-Conference.pdf.

Li Lin and Xiaojun Wan. ScaleSweep: Accurate NVFP4 Post-Training Quantization of LLMs via Block Scale Initialization, 2026. URL https://arxiv.org/abs/2606.07618.

Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. LLM-QAT: Data-Free Quantization Aware Training for Large Language Models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 467–484, August 2024. doi: 10.18653/v1/2024.findings-acl.26. URL https://aclanthology.org/2024.findings-acl.26/.

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*, volume 2025, pp. 92009–92032, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/e5b1c0d4866f72393c522c8a00eed4eb-Paper-Conference.pdf.

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, July 2026. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.388. URL https://aclanthology.org/2026.acl-long.388/.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer Sentinel Mixture Models, 2016. URL https://arxiv.org/abs/1609.07843.

Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or Down? Adaptive Rounding for Post-Training Quantization. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20, 2020.

NVIDIA. Introducing NVFP4 for Efficient and Accurate Low-Precision Inference. NVIDIA Technical Blog, 2025. URL https://developer.nvidia.com/blog/?p=102000.

NVIDIA. NVIDIA TensorRT Model Optimizer Documentation. Online documentation, 2026. URL https://nvidia.github.io/Model-Optimizer/.

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 (Volume 1: Long Papers)*, pp. 1525–1534, August 2016. doi: 10.18653/v1/P16-1144. URL https://aclanthology.org/P16-1144/.

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. URL http://jmlr.org/papers/v21/20-074.html.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106, August 2021. ISSN 0001-0782. doi: 10.1145/3474381. URL https://doi.org/10.1145/3474381.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Gao Peng, Yu Qiao, and Ping Luo. OmniQuant: Omnidirectionally Calibrated Quantization for Large Language Models. In *International Conference on Learning Representations*, volume 2024, pp. 45472–45496, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/c6483c8a68083af3383f91ee0dc6db95-Paper-Conference.pdf.

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 *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38087–38099, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/xiao23c.html.

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.

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, July 2019. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

# APPENDIX CONTENTS


| SECTION | | | PAGE |
|---|---|---|---|
| **A** | **NVFP4 Format and Channel Clusters** | | 16 |
| **B** | **Weight-Only and Activation-Only Analysis** | | 17 |
| **C** | **Additional DACC Details** | | 18 |
| | C.1 | DACC Clustering Procedure | 18 |
| | C.2 | Sensitivity to Weight-Aware Exponent | 18 |
| **D** | **BITR Accuracy and Calibration Cost** | | 20 |
| **E** | **Mechanism Analysis of DACC and BITR** | | 21 |
| **F** | **Transformation Placement** | | 23 |
| **G** | **Detailed Deployment Results** | | 24 |
| **H** | **ABSO Iteration Analysis** | | 25 |
| **I** | **Downstream-Task Results and Evaluation Protocol** | | 26 |
| | I.1 | Detailed Results | 26 |
| | I.2 | Evaluation Protocol | 27 |
| **J** | **Evaluation across Model Scales** | | 28 |
| **K** | **Calibration Sensitivity** | | 29 |
| | K.1 | Sensitivity to the Calibration Dataset | 29 |
| | K.2 | Sensitivity to Calibration Size | 29 |
| **L** | **Compatibility with Existing Quantization Methods** | | 30 |
| **M** | **Robustness to Random Seeds** | | 31 |

# A NVFP4 Format and Channel Clusters

Figure 3 illustrates two basic concepts. First, NVFP4 applies one FP32 tensor scale to the entire matrix, while every block contains 16 E2M1 values sharing one E4M3 block scale. Second, the activation and weight matrices share the input-channel dimension $C_i$. For an activation matrix, the 16 consecutive $C_i$ elements at a fixed token form one activation block. For a weight matrix, the corresponding 16 $C_i$ elements at a fixed output channel form one weight block. Each block uses its own block scale. We refer to the same set of 16 consecutive input channels across tokens as a *channel cluster*.

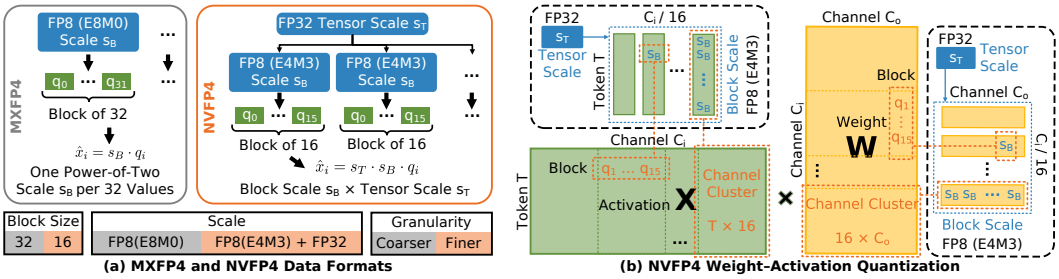


Figure 3: (a) Visualization comparing the NVFP4 and MXFP4 data formats. (b) Visualization of a channel cluster defined in this work, consisting of 16 consecutive channels along the input-channel dimension $C_i$ shared by the activation and weight matrices.

## B Weight-Only and Activation-Only Analysis

To isolate the source of degradation after rotation, we separately evaluate weight-only (W4A16) and activation-only (W16A4) quantization. Table 4 shows that MR-GPTQ generally improves weight-only quantization over GPTQ, whereas it consistently degrades activation-only quantization. This indicates that its weight compensation remains effective, while the transformed activations are the main source of degradation under NVFP4.

For the W16A4 setting, GPTQ performs no operation on the activations. Since the weights remain in 16-bit precision, this setting is equivalent to RTN for activation quantization, resulting in identical values.

Table 4: Weight-only and activation-only perplexity. BF16 is shown as the unquantized BF16 reference. The best quantized result in each setting is shown in bold. Lower is better.

| Setting | Method | SmolLM2-135M | | Qwen3-0.6B | | Llama3.2-1B | | Qwen3-1.7B | |
|---|---|---|---|---|---|---|---|---|---|
| | | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 |
| BF16 | NONE | 15.444 | 21.355 | 20.957 | 30.329 | 9.750 | 14.016 | 16.719 | 22.394 |
| W4A16 | RTN | 17.985 | 24.774 | 23.563 | 33.454 | 10.663 | 15.527 | 18.163 | 23.331 |
| | GPTQ | 17.667 | 24.071 | 23.352 | 32.619 | 10.320 | **14.909** | 17.365 | 23.210 |
| | MR-GPTQ | **16.966** | **23.153** | **23.325** | **32.098** | **10.304** | 14.942 | **17.160** | **22.851** |
| W16A4 | RTN | **17.943** | **24.424** | **23.190** | **32.885** | **10.417** | **15.154** | **17.245** | **23.064** |
| | GPTQ | **17.943** | **24.424** | **23.190** | **32.885** | **10.417** | **15.154** | **17.245** | **23.064** |
| | MR-GPTQ | 19.289 | 26.061 | 23.926 | 34.421 | 10.552 | 15.357 | 17.382 | 23.121 |

# C ADDITIONAL DACC DETAILS

Algorithm C.1 computes the pairwise proxy distortion used by DACC before the complete channel clusters and their true block scales are known. The $10^{-10}$ scale floor prevents numerical instability for zero or extremely small channel values; it does not change any nonzero pair.

**Algorithm C.1** Pairwise proxy distortion

**Require:** Channel profiles $\mathbf{x}_a, \mathbf{x}_b \in \mathbb{R}^T$
**Ensure:** Proxy distortion $D$
$D \leftarrow 0$
**for** $t = 1, \ldots, T$ **do**
  $M \leftarrow \max(|x_{t,a}|, |x_{t,b}|)$ ▷ proxy block maximum
  $m \leftarrow \min(|x_{t,a}|, |x_{t,b}|)$ ▷ value being evaluated
  $s \leftarrow \max(M/6, 10^{-10})$ ▷ map $M$ to E2M1 value 6
  $q^\star \leftarrow \arg\min_{q \in \mathcal{C}_{\mathrm{E2M1}}} |m - sq|$ ▷ nearest code point
  $D \leftarrow D + (m - sq^\star)^2$ ▷ accumulate token distortion
**end for**
**return** $D$

## C.1 DACC CLUSTERING PROCEDURE

Algorithm C.2 summarizes the complete incremental clustering procedure used by DACC. It maintains cumulative seed distances and cluster maximum envelopes, constructs the permutation once during calibration, and performs no iterative reassignment.

**Algorithm C.2** DACC channel clustering

**Require:** Calibration activations $X$, weight $W$, block size $B$, exponent $\alpha$
**Ensure:** Channel permutation $\mathbf{p}$
$C \leftarrow$ number of channels; $K \leftarrow C/B$ ▷ $K$ target clusters
**for** $c = 1, \ldots, C$ **do** ▷ *Stage 1: select a stable initial seed*
  $I_c \leftarrow \|W_{:,c}\|_2^\alpha \sum_t |X_{t,c}|$ ▷ weight-aware importance
**end for**
$s_1 \leftarrow \arg\max_c I_c$ ▷ first stable seed
$\mathcal{S} \leftarrow \{s_1\}$
**for all** $c \notin \mathcal{S}$ **do**
  $D_c \leftarrow \|W_{:,c}\|_2^\alpha \textsc{ProxyDist}(X_{:,c}, X_{:,s_1})$ ▷ initialize cumulative distance
**end for**
**while** $|\mathcal{S}| < K$ **do**
  $c^\star \leftarrow \arg\max_{c \notin \mathcal{S}} D_c$ ▷ largest cumulative distortion
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{c^\star\}$ ▷ place in a new cluster
  **for all** $c \notin \mathcal{S}$ **do**
    $D_c \leftarrow D_c + \|W_{:,c}\|_2^\alpha \textsc{ProxyDist}(X_{:,c}, X_{:,c^\star})$ ▷ incremental update
  **end for**
**end while**
Initialize $\mathcal{G}_j = \{s_j\}$ and $\mathbf{m}_j = |X_{:,s_j}|$ ▷ *Stage 3: greedily fill;* cluster maximum envelopes
Sort unselected channels by decreasing $I_c$ ▷ stabilize envelopes early
**for all** unselected channels $c$ **do**
  $j^\star \leftarrow \arg\min_{j:\,|\mathcal{G}_j|<B} \textsc{ProxyDist}(X_{:,c}, \mathbf{m}_j)$
  $\mathcal{G}_{j^\star} \leftarrow \mathcal{G}_{j^\star} \cup \{c\}$ ▷ choose most compatible cluster
  $\mathbf{m}_{j^\star} \leftarrow \max(\mathbf{m}_{j^\star}, |X_{:,c}|)$ ▷ refresh maximum envelope
**end for**
$\mathbf{p} \leftarrow \mathrm{Concat}(\mathcal{G}_1, \ldots, \mathcal{G}_K)$ ▷ make clusters contiguous
**return** $\mathbf{p}$

## C.2 SENSITIVITY TO WEIGHT-AWARE EXPONENT

Table 5 reports the sensitivity to the weight-aware exponent $\alpha$ on Qwen3-1.7B. The results remain stable over a broad range of values. We use $\alpha = 2$, which achieves the lowest average perplexity among the evaluated settings and does not require dataset-specific tuning.

All evaluated linear dimensions are divisible by the NVFP4 block size $B = 16$, as assumed by our implementation. DACC is executed once during calibration and performs no runtime clustering or iterative reassignment. Pairwise proxy distances are accumulated incrementally rather than stored as a complete $C \times C$ matrix, giving a calibration-time complexity of $O(TC^2/B)$.

Table 5: Sensitivity of DACC to $\alpha$ on Qwen3-1.7B. Avg. denotes the mean perplexity over WikiText-2 and C4. "No DACC" denotes disabling DACC while keeping the other settings unchanged.

| $\alpha$ | No DACC | 0.0 | 0.5 | 1.0 | 1.5 | 2.0 | 2.5 | 3.0 | 3.5 | 4.0 |
|---|---|---|---|---|---|---|---|---|---|---|
| WKT2 | 18.008 | 17.210 | **17.076** | 17.840 | 17.541 | 17.120 | 17.136 | 17.293 | 17.296 | 17.620 |
| C4 | 23.911 | 23.525 | 23.309 | 23.426 | 23.536 | **23.208** | 23.268 | 23.296 | 23.298 | 23.317 |
| Avg. | 20.960 | 20.367 | 20.192 | 20.633 | 20.538 | **20.164** | 20.202 | 20.294 | 20.297 | 20.468 |

## D BITR ACCURACY AND CALIBRATION COST

We compare BITR with the full-channel AWQ-style rescaling used by Four-Over-Six. Table 6 evaluates quantization accuracy and offline preprocessing cost on Qwen3-8B. With three search rounds, BITR ($k = 5$) achieves an average perplexity close to full-channel AWQ-style rescaling (16.073 versus 16.067), while reducing the complete preprocessing time from 68.3 to 31.2 minutes, a 54.3% reduction. Relative to the no-rescaling configuration (23.5 minutes), BITR adds 7.7 minutes to the preprocessing pipeline, compared with 44.8 minutes for full-channel search. The broader sweep supports $k = 5$ and three rounds as our default configuration.

Table 6: BITR design choices and comparison with full-channel AWQ-style rescaling on Qwen3-8B. Time includes the complete preprocessing pipeline. For BITR, $k = 16$ includes all channels within each block.

| Method | Top-$k$ | Rounds | WKT2 | C4 | Avg. | Time (min) |
|---|---|---|---|---|---|---|
| No rescaling | – | – | 12.371 | 19.805 | 16.088 | 23.5 |
| AWQ-style | All | 1 | 12.430 | 19.809 | 16.120 | 43.3 |
| AWQ-style | All | 3 | 12.385 | 19.748 | **16.067** | 68.3 |
| BITR | 3 | 1 | 12.432 | 19.808 | 16.120 | 27.3 |
| BITR | 3 | 3 | 12.585 | 19.936 | 16.261 | 29.1 |
| BITR | 3 | 5 | 12.498 | 19.902 | 16.200 | 31.2 |
| BITR | 5 | 1 | 12.409 | 19.933 | 16.171 | 27.9 |
| BITR | 5 | 3 | 12.396 | 19.751 | **16.073** | 31.2 |
| BITR | 5 | 5 | 12.420 | 19.786 | 16.103 | 34.3 |
| BITR | 10 | 1 | 12.427 | 19.849 | 16.138 | 29.8 |
| BITR | 10 | 3 | 12.348 | 19.847 | 16.097 | 35.8 |
| BITR | 10 | 5 | 12.411 | 19.789 | 16.100 | 42.5 |
| BITR | 16 (all) | 1 | 12.411 | 19.837 | 16.124 | 31.4 |
| BITR | 16 (all) | 3 | 12.400 | 19.898 | 16.149 | 41.8 |
| BITR | 16 (all) | 5 | 12.325 | 19.832 | 16.079 | 52.0 |

# E MECHANISM ANALYSIS OF DACC AND BITR

We examine two complementary questions: whether DACC improves distortion-aware channel clustering, and whether BITR further reduces joint weight-activation reconstruction errors under the resulting layout. The first analysis compares permutation-only methods using both global and regional distortion metrics. The second evaluates the incremental effect of BITR across four models.

**Distortion-aware clustering with DACC.** We collect calibration activations from layers 0, 15, and 31 of Llama3-8B and analyze four projection-input groups: QKV, O, Gate/Up, and Down. For each input matrix, at most 4,096 activation tokens are retained.

In addition to the original, random, and magnitude-sorted layouts, we include two stronger heuristic controls. Outlier Staggering greedily separates channels whose activation and weight outliers frequently co-occur. Joint 2nd-Moment sorts channels according to the joint activation-weight second-moment criterion used by PermuQuant (Cheng et al., 2026), with $\alpha = 0.5$. It implements only the ordering criterion, without the calibration-based acceptance rule of the complete PermuQuant method.

Activation MSE is measured in the original activation domain. MDR and HDR denote the moderate- and high-distortion regions defined in Section 3. For a region $\mathcal{R}$, its absolute regional MSE is $\sum_{i\in\mathcal{R}} e_i/N$, where $e_i$ is the squared reconstruction error and $N$ is the total number of activation values. It is therefore directly comparable with the global MSE and additive across regions. Results are averaged over layers 0, 15, and 31 and the four projection inputs.

Table 7: Effect of channel ordering on NVFP4 activation distortion in representative Llama3-8B layers. All methods in this table modify only the channel order. MSE-style quantities are reported in units of $10^{-4}$. Regional MSE columns report absolute contributions to the global activation MSE. The Random row averages five independent permutations. Column minima are boldfaced.

| Method | MSE ↓ | MDR (%) | HDR (%) | MDR+HDR MSE ↓ | HDR MSE ↓ |
|---|---|---|---|---|---|
| Original | 7.323 | 13.440 | 4.210 | 3.914 | 1.943 |
| Random | 7.349 | 13.456 | 4.225 | 3.941 | 1.960 |
| Magnitude | 8.192 | 16.823 | 5.672 | 5.578 | 2.887 |
| Outlier Staggering | 7.183 | **12.807** | 3.842 | 3.625 | 1.737 |
| Joint 2nd-Moment | 8.157 | 16.560 | 5.481 | 5.568 | 2.884 |
| DACC | **6.866** | 13.019 | **3.826** | **3.509** | **1.632** |

As shown in Table 7, random permutation remains close to the original layout. Across five random permutations, the standard deviations of global MSE, MDR+HDR absolute MSE, and HDR absolute MSE are only $0.009$, $0.019$, and $0.016$, respectively, in units of $10^{-4}$. DACC improves these quantities over the Random mean by $0.483$, $0.432$, and $0.328$, far beyond random-layout variation. Thus, its benefit does not follow from arbitrary channel permutation.

Magnitude ordering and joint second-moment ordering substantially increase both total MSE and regional MSE, showing that clustering channels solely by statistical magnitude does not address the non-uniform NVFP4 codebook. Outlier Staggering is a useful control: it obtains the lowest MDR occupancy and reduces total MSE relative to the original layout. Nevertheless, DACC achieves the lowest total MSE among all permutation-only methods, reducing it by $6.24\%$ relative to the original layout and by $4.42\%$ relative to Outlier Staggering. More importantly, DACC reduces MDR+HDR absolute MSE by $10.35\%$ and HDR absolute MSE by $16.00\%$ relative to the original layout. It also attains the lowest HDR occupancy. These results directly support the distortion-aware clustering criterion used by DACC.

**Cross-model incremental effect of BITR.** We further evaluate the incremental effect of BITR under the DACC layout on Llama-3.2-1B, Qwen3-1.7B, Llama3-8B, and Qwen3-8B. For a model with $L$ Transformer layers, we select layers $0$, $\lfloor (L-1)/2 \rfloor$, and $L-1$. We evaluate four projection-input groups in each selected block: QKV, O, Gate/Up, and Down. Q/K/V share the same input activations, as do Gate/Up, so each shared input is evaluated once.

Activation MSE is measured in the original activation units, undoing BITR's channel scaling after reconstruction. We additionally measure joint weight-activation reconstruction error as the output MSE between $\hat{X}\hat{W}^\top$, formed from the quantized activations and weights, and the FP32 reference $XW^\top$. Q/K/V and Gate/Up weights are concatenated within their respective projection groups. This local W4A4 analysis uses the same weight quantizer for both configurations, without GPTQ compensation or ABSO.

Table 8: Incremental effect of BITR under the DACC layout. Activation and output MSEs are averaged over the selected layer-projection groups. Reductions are relative to DACC and computed before rounding. The final row reports the mean of model-wise reductions.

| Model | Activation MSE ↓ | | | Output MSE ↓ | | |
|---|---|---|---|---|---|---|
| | DACC | +BITR | Red. (%) | DACC | +BITR | Red. (%) |
| Llama3.2-1B | $6.55\times10^{-4}$ | $\mathbf{5.88\times10^{-4}}$ | 10.27 | $1.69\times10^{-3}$ | $\mathbf{1.54\times10^{-3}}$ | 9.17 |
| Qwen3-1.7B | $1.31\times10^{-1}$ | $\mathbf{9.89\times10^{-2}}$ | 24.63 | 1.30 | $\mathbf{6.96\times10^{-1}}$ | 46.49 |
| Llama3-8B | $6.84\times10^{-4}$ | $\mathbf{5.93\times10^{-4}}$ | 13.25 | $1.66\times10^{-3}$ | $\mathbf{1.40\times10^{-3}}$ | 15.42 |
| Qwen3-8B | $7.08\times10^{-3}$ | $\mathbf{5.58\times10^{-3}}$ | 21.15 | $7.95\times10^{-2}$ | $\mathbf{4.30\times10^{-2}}$ | 45.95 |
| Average | – | – | **17.32** | – | – | **29.26** |

As shown in Table 8, adding BITR reduces activation MSE by $10.27\%$-$24.63\%$ and output MSE by $9.17\%$-$46.49\%$ across the four evaluated models. The average model-wise reductions are $17.32\%$ and $29.26\%$, respectively. These results show that bounded channel rescaling improves not only activation reconstruction but also output reconstruction under joint weight-activation quantization. Together, the two analyses characterize DACC's distortion-aware clustering and BITR's additional reconstruction benefit.

# F TRANSFORMATION PLACEMENT

DACC and BITR are applied to the inputs of the QKV, O, Gate/Up, and Down projections. Their deployment locations fall into the following three categories.

**Materialized offline in weights.** For every transformed linear layer, the same channel permutation and inverse rescaling are applied to its weight before quantization: $W \mapsto WPR^{-1}$, where $P$ and $R$ denote the DACC permutation and BITR rescaling, respectively. The transformed weight is materialized once during model export, so no weight-side transformation is performed during inference.

**Absorbed between adjacent operators.** In the evaluated model graph, the DACC permutation of the MLP intermediate dimension is absorbed between the Gate/Up and Down projections. Applying the same permutation to the Gate and Up outputs preserves the element-wise SiLU and multiplication operations, while the inverse permutation is incorporated into the Down-projection weight. Therefore, this permutation introduces no standalone runtime operator. A similar V-to-O absorption is algebraically possible for standard multi-head attention when the corresponding dimensions match, but it is not used in our evaluated GQA models.

**Retained online in the evaluated GQA deployment.** The activation-side transformations at the QKV and Gate/Up inputs cannot generally be absorbed across the preceding normalization and residual boundaries. Moreover, the V-to-O permutation is retained online because the query-head and key/value-head dimensions differ under GQA. BITR activation rescaling is also executed online at the applicable projection inputs, and ABSO remains online because its block scales depend on runtime activations. These operations are placed adjacent to activation quantization and are included in both the controlled Transformers evaluation in Table 3 and the vLLM deployment evaluation in Table 9.

## G Detailed Deployment Results

**vLLM deployment.** The vLLM experiment evaluates BF16 and four packed NVFP4 implementations under identical engine settings. Prefix caching is disabled and CUDA Graph is enabled for every method. Each model is evaluated in a fresh process to avoid sharing CUDA allocator state, model weights, or KV-cache storage across methods. Table 9 reports the complete results.

Table 9: Detailed end-to-end vLLM results on Qwen3-8B. Throughput is measured in output tokens/s and peak memory in GiB. Generation-stage peak GPU memory is sampled via NVML at 20-ms intervals. Each entry is the median of five runs after two warm-up runs.

**(a) Output-token throughput (tokens/s) ↑**

| Batch size | BF16 | RTN | GPTQ | MR-GPTQ | NADA |
|---|---|---|---|---|---|
| 1 | 90.43 | 105.44 | 105.54 | 105.34 | 102.34 |
| 2 | 171.42 | 199.84 | 200.37 | 199.90 | 192.11 |
| 4 | 314.48 | 389.73 | 390.39 | 390.05 | 377.93 |
| 8 | 566.68 | 762.87 | 762.42 | 763.09 | 704.28 |
| 16 | 957.44 | 1370.28 | 1370.25 | 1372.94 | 1205.16 |

**(b) Peak GPU memory (GiB) ↓**

| Batch size | BF16 | RTN | GPTQ | MR-GPTQ | NADA |
|---|---|---|---|---|---|
| 1 | 25.64 | 10.00 | 10.00 | 10.00 | 10.10 |
| 2 | 25.64 | 10.01 | 10.02 | 10.00 | 10.10 |
| 4 | 25.81 | 10.12 | 10.12 | 10.11 | 10.21 |
| 8 | 26.14 | 10.46 | 10.46 | 10.45 | 10.66 |
| 16 | 26.85 | 11.17 | 11.17 | 11.16 | 11.20 |

NADA is faster than BF16 across all evaluated batch sizes. Relative to the fastest quantized baseline, its throughput difference ranges from 3.0% to 12.2%. This gap includes all online DACC permutations, BITR rescaling, and ABSO block-scale updates. Peak memory remains close to the other NVFP4 methods, with a maximum difference of 0.21 GiB.

**Controlled Transformers comparison.** The official Four-Over-Six implementation quantizes each weight once with its CUDA MSE scale rule and applies the same rule to every runtime activation. NADA instead loads its calibrated packed weights and invokes a fused CUDA activation kernel. The kernel jointly applies the stored channel permutation and rescaling, computes the single-step ABSO scale, converts it to E4M3, packs the E2M1 values, and writes the result directly in the CUTLASS scale-factor layout. Both paths subsequently invoke the same CUTLASS NVFP4 matrix multiplication.

## H ABSO ITERATION ANALYSIS

**Why one update is sufficient.** Table 10 separates the effect of repeated ABSO updates with and without the preceding DACC and BITR preprocessing. After DACC and BITR, increasing ABSO from one to three updates worsens the average perplexity from 22.054 to 22.118, indicating that three updates provide no further improvement in the representative setting. Without DACC and BITR, however, three updates still reduce the average perplexity from 24.264 to 24.213. These results support using a single ABSO update after DACC and BITR in this setting.

Table 10: Effect of preceding activation preprocessing on the required number of ABSO updates for SmolLM2-135M. Avg. is the mean perplexity over the two corpora. Lower is better.

| Preprocessing | ABSO updates | WKT2 | C4 | Avg. |
|---|---|---|---|---|
| DACC + BITR | 1 | **18.841** | **25.266** | **22.054** |
| | 3 | 18.849 | 25.387 | 22.118 |
| None | 1 | 20.623 | **27.905** | 24.264 |
| | 3 | **20.503** | 27.923 | **24.213** |

# I Downstream-Task Results and Evaluation Protocol

This section presents the complete downstream-task results and the corresponding evaluation settings for the four models used in the main comparison.

## I.1 Detailed Results

Table 11 reports performance on ten benchmarks covering commonsense reasoning and reading comprehension (PIQA, WinoGrande, BoolQ, and HellaSwag), knowledge, scientific reasoning, and contextual word prediction (MMLU, ARC-Easy, LAMBADA, and ARC-Challenge), and mathematical reasoning and code generation (GSM8K and MBPP). All scores are expressed as percentages and rounded to two decimal places. Higher scores indicate better performance. Task-specific metrics and evaluation settings are provided in Appendix I.2.

Table 11: Detailed downstream-task performance across four models and ten benchmarks. ARC-E and ARC-C denote ARC-Easy and ARC-Challenge, respectively. Best quantized results in each row are boldfaced.

| Model | Task | BF16 | RTN | GPTQ | MR-GPTQ | 4/6 + AWQ | ARCQuant | NADA |
|---|---|---|---|---|---|---|---|---|
| Llama3.2-1B | PIQA | 74.70 | 71.82 | 72.85 | 72.03 | 71.71 | 72.47 | **73.34** |
| | WinoGrande | 62.59 | 60.54 | 60.62 | 60.22 | 60.30 | 57.77 | **61.40** |
| | BoolQ | 63.98 | 58.78 | 61.35 | 62.75 | 59.63 | 59.17 | **63.64** |
| | HellaSwag | 59.55 | 56.45 | 56.73 | 56.20 | 57.00 | 56.05 | **57.67** |
| | MMLU | 31.75 | 26.63 | 28.35 | 27.28 | 28.44 | 27.52 | **29.17** |
| | ARC-E | 71.34 | 67.26 | 68.14 | 67.26 | 67.26 | **69.15** | 67.72 |
| | LAMBADA | 62.12 | 53.48 | 57.23 | **58.78** | 54.20 | 57.54 | 58.74 |
| | ARC-C | 39.85 | 36.18 | 35.92 | 35.84 | 35.41 | 37.46 | **37.54** |
| | GSM8K | 5.76 | 3.56 | **4.25** | 3.18 | 4.17 | 2.88 | **4.25** |
| | MBPP | 27.40 | 16.80 | 16.80 | 15.20 | 16.80 | 14.40 | **17.60** |
| Qwen3-1.7B | PIQA | 72.20 | 69.69 | 69.59 | 69.37 | 69.42 | 69.48 | **70.29** |
| | WinoGrande | 60.85 | 59.20 | 58.25 | 59.35 | 57.62 | **60.30** | 59.75 |
| | BoolQ | 77.86 | 73.55 | 76.18 | **76.85** | 74.95 | 73.30 | 72.11 |
| | HellaSwag | 56.50 | 55.02 | 54.55 | 55.05 | **55.90** | 55.10 | 55.88 |
| | MMLU | 62.82 | 58.50 | 58.79 | 58.13 | 57.84 | 58.10 | **59.59** |
| | ARC-E | 79.63 | 77.15 | 77.31 | 76.68 | 76.85 | 75.88 | **78.37** |
| | LAMBADA | 50.44 | 45.57 | 47.33 | 45.61 | 44.40 | 44.29 | **47.82** |
| | ARC-C | 53.07 | 48.55 | 49.40 | 48.63 | 46.84 | 48.46 | **49.91** |
| | GSM8K | 68.76 | 55.19 | 57.39 | 57.24 | 54.59 | 58.53 | **62.77** |
| | MBPP | 42.40 | 21.00 | 26.60 | 33.00 | **35.60** | 25.00 | 35.40 |
| Llama3-8B | PIQA | 80.58 | 78.73 | 79.38 | 78.89 | 79.05 | 78.78 | **79.49** |
| | WinoGrande | 78.22 | 75.30 | 76.01 | 75.77 | 74.98 | **76.48** | 75.93 |
| | BoolQ | 82.02 | 81.19 | 80.98 | 80.67 | 76.79 | 76.48 | **81.80** |
| | HellaSwag | 73.92 | 71.92 | 72.15 | **72.55** | 71.75 | 71.17 | 72.42 |
| | MMLU | 67.80 | 64.37 | 64.76 | 64.44 | 64.61 | 64.36 | **65.38** |
| | ARC-E | 85.23 | 84.01 | 83.46 | 84.05 | 83.96 | 83.33 | **84.64** |
| | LAMBADA | 75.49 | 74.23 | **75.16** | 73.59 | 74.60 | 73.43 | 74.33 |
| | ARC-C | 58.19 | 56.06 | **57.85** | 55.97 | 56.31 | 55.72 | 56.74 |
| | GSM8K | 49.36 | 37.98 | 39.58 | 38.59 | 38.44 | 38.97 | **39.88** |
| | MBPP | 49.80 | 43.60 | 42.20 | **45.00** | 43.80 | 44.20 | 41.40 |
| Qwen3-8B | PIQA | 74.05 | 72.09 | 73.07 | **73.72** | 73.29 | 73.01 | 73.34 |
| | WinoGrande | 67.64 | 66.53 | 67.17 | 64.72 | 66.54 | 66.06 | **68.35** |
| | BoolQ | 86.21 | 85.17 | 85.63 | **85.75** | 85.44 | 85.38 | 85.66 |
| | HellaSwag | 63.38 | **62.65** | 61.60 | 62.28 | 62.10 | 60.68 | 61.75 |
| | MMLU | 75.04 | 72.68 | 72.69 | 72.58 | 72.61 | 72.73 | **73.12** |
| | ARC-E | 82.41 | 81.23 | **82.11** | 81.82 | 81.57 | 81.82 | 81.78 |
| | LAMBADA | 54.26 | 52.47 | **54.34** | 52.47 | 52.50 | 51.70 | 53.25 |
| | ARC-C | 60.24 | 59.30 | **59.47** | 58.28 | 58.70 | 58.19 | 58.87 |
| | GSM8K | 82.26 | **80.59** | 79.38 | 78.17 | 76.80 | 78.92 | 79.38 |
| | MBPP | 42.00 | 37.40 | 33.80 | **39.00** | 37.60 | 35.80 | 38.80 |

### I.2 Evaluation Protocol

All methods evaluated on a given model use the same tokenizer, evaluation examples, few-shot configuration, and maximum context length of 2,048 tokens. No chat template is applied. Table 12 summarizes the task-specific few-shot settings, evaluation sizes, and metrics. The HellaSwag subset and the MMLU per-subject limit are applied identically to BF16 and all quantized methods.

Table 12: Downstream-task evaluation protocol. "Full" denotes the complete available evaluation split. Metric names correspond to the output keys of LM Evaluation Harness.

| Task | Few-shot | Evaluation size | Metric |
|---|---|---|---|
| PIQA | 0 | Full | `acc_norm` |
| WinoGrande | 5 | Full | `acc` |
| BoolQ | 0 | Full | `acc` |
| HellaSwag | 10 | 4,000-example subset | `acc_norm` |
| MMLU | 5 | $\leq 200$ per subject | `acc` |
| ARC-Easy | 25 | Full | `acc_norm` |
| ARC-Challenge | 25 | Full | `acc_norm` |
| LAMBADA | 0 | Full | `acc` |
| GSM8K | 5 | Full | `exact_match` |
| MBPP | 3 | Full | `pass@1` |

## J Evaluation across Model Scales

To evaluate NADA over a broader range of model sizes, we additionally test two compact models, SmolLM2-135M and Qwen3-0.6B, and two large models, Qwen3-32B and Llama3-70B. Table 13 reports the WikiText-2 and C4 perplexities. However, ARCQuant supports only specific matrix dimensions in its current implementation, preventing its application to the compact models. All perplexity evaluations are conducted on NVIDIA RTX 5090D GPUs, except for Qwen3-32B and Llama3-70B, which are evaluated on an NVIDIA RTX PRO 6000 GPU due to their larger memory requirements.

Table 13: NVFP4 W4A4 perplexity across model scales. Lower is better. The best quantized result in each column is boldfaced. "OOM" indicates an out-of-memory failure.

| | SmolLM2-135M | | Qwen3-0.6B | | Qwen3-32B | | Llama3-70B | |
|---|---|---|---|---|---|---|---|---|
| Method | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 |
| BF16 | 15.444 | 21.355 | 20.957 | 30.329 | 7.651 | 12.518 | 2.856 | 7.165 |
| RTN | 21.389 | 28.930 | 26.205 | 36.558 | 8.036 | 13.016 | 3.618 | 8.317 |
| GPTQ | 21.118 | 28.148 | 25.680 | 35.745 | 7.883 | 12.850 | 3.691 | 7.836 |
| MR-GPTQ | 21.906 | 29.358 | 26.846 | 36.346 | 7.924 | 12.842 | 3.820 | 7.882 |
| 4/6 + AWQ | 21.241 | 28.726 | 26.101 | 37.471 | 8.001 | 12.998 | OOM | OOM |
| ARCQuant | – | – | – | – | 8.025 | 13.018 | **3.546** | 7.801 |
| NADA | **18.841** | **25.266** | **23.146** | **33.428** | **7.851** | **12.785** | 3.567 | **7.670** |

The results show that NADA remains effective on the evaluated models from 135M to 70B parameters. Larger evaluated models generally exhibit smaller relative degradation under NVFP4 quantization. However, model size is confounded with architecture and model family in this comparison; therefore, this observation should be interpreted as a cross-model trend rather than a strict scaling law.

On Llama3-70B, NADA obtains the best C4 perplexity but is slightly worse than ARCQuant on WikiText-2. GPTQ, which serves as the default weight-quantization backbone of NADA, also underperforms RTN on WikiText-2 for this model and may contribute to this result. However, the current experiments do not identify the cause of the remaining gap, and we do not attribute it conclusively to calibration specialization. Evaluation with additional calibration corpora at the 70B scale is left for future work.

# K CALIBRATION SENSITIVITY

We further study the robustness of NADA with respect to the calibration dataset and calibration-data size. Unless otherwise specified, our default setting uses 128 C4 sequences with a sequence length of 2,048.

## K.1 SENSITIVITY TO THE CALIBRATION DATASET

A quantization method that strongly overfits its calibration corpus may degrade when the calibration and evaluation domains differ. We therefore calibrate NADA using FineWeb-Edu, C4, and WikiText-2, while retaining the same WikiText-2 and C4 evaluation protocol. Table 14 reports the results on models ranging from 1B to 8B parameters.

Table 14: Perplexity obtained using different calibration datasets. Lower is better. The best NADA result in each column is boldfaced.

| | Llama3.2-1B | | Qwen3-1.7B | | Llama3-8B | | Qwen3-8B | |
|---|---|---|---|---|---|---|---|---|
| Calibration set | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 |
| BF16 | 9.750 | 14.016 | 16.719 | 22.394 | 6.137 | 9.446 | 11.994 | 19.197 |
| FineWeb-Edu | 10.986 | 16.053 | 17.993 | 23.632 | 6.689 | 10.495 | 12.351 | 19.819 |
| C4 | 10.996 | **15.985** | **17.120** | **23.208** | 6.710 | **10.440** | 12.396 | **19.751** |
| WKT2 | **10.770** | 16.161 | 17.289 | 23.676 | **6.633** | 10.512 | **12.330** | 19.886 |

Across the three calibration corpora, NADA exhibits limited calibration-corpus sensitivity on WikiText-2 and C4 perplexity. Calibration on WikiText-2 generally favors WikiText-2 evaluation, whereas C4 calibration usually produces the best C4 perplexity, indicating a mild domain preference. FineWeb-Edu remains competitive despite its domain difference, although Qwen3-1.7B shows greater variation than the other models. These results characterize sensitivity within the evaluated corpora and metrics, rather than establishing the absence of calibration overfitting in other domains.

## K.2 SENSITIVITY TO CALIBRATION SIZE

We next vary the number and sequence length of calibration samples. Table 15 covers 64-256 sequences and sequence lengths of 1,024 and 2,048. All other quantization settings remain unchanged.

Table 15: Sensitivity to the number and length of calibration sequences. All results are perplexities, and lower is better. The best result in each column is boldfaced.

| Length | Samples | Llama3.2-1B | | Qwen3-1.7B | | Llama3-8B | | Qwen3-8B | |
|---|---|---|---|---|---|---|---|---|---|
| | | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 |
| 1,024 | 64 | 12.537 | 16.588 | 19.860 | 24.721 | 7.517 | 10.635 | 14.129 | 20.939 |
| | 128 | 12.394 | 16.547 | 19.071 | 24.312 | 7.497 | 10.546 | 14.136 | 20.796 |
| | 256 | 12.408 | 16.476 | 19.213 | 24.354 | 7.461 | 10.494 | 14.111 | 20.858 |
| 2,048 | 64 | 10.960 | 15.998 | 17.319 | **23.047** | 6.732 | 10.457 | 12.416 | 19.769 |
| | 128 | 10.996 | 15.985 | **17.120** | 23.208 | 6.711 | 10.441 | **12.396** | 19.751 |
| | 256 | **10.954** | **15.905** | 17.330 | 23.433 | **6.707** | **10.399** | 12.409 | **19.750** |

Increasing the sequence length from 1,024 to 2,048, and therefore increasing the number of calibration tokens at a fixed sample count, consistently improves perplexity. In contrast, varying the number of samples from 64 to 256 at a fixed sequence length produces relatively small and non-monotonic changes. These results indicate that NADA is not sensitive to the exact number of calibration sequences once a moderate calibration budget is available. The default configuration of 128 sequences with length 2,048 provides a favorable balance between calibration cost and accuracy.

## L Compatibility with Existing Quantization Methods

We further evaluate whether NADA can be combined with different quantization backbones and activation preprocessing methods. Specifically, we replace the default GPTQ backbone with RTN, or apply the MicroRot (MR) preprocessing from MR-GPTQ or SmoothQuant before all NADA preprocessing stages. The results are reported in Table 16.

Table 16: NADA variants with different quantization backbones and preprocessing methods under NVFP4 W4A4 quantization.

| | Llama3.2-1B | | Qwen3-1.7B | | Llama3-8B | | Qwen3-8B | |
|---|---|---|---|---|---|---|---|---|
| Method | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 |
| BF16 | 9.750 | 14.016 | 16.719 | 22.394 | 6.137 | 9.446 | 11.994 | 19.197 |
| NADA (RTN) | 11.343 | 16.740 | 18.528 | 24.336 | 6.808 | 10.562 | 12.639 | 20.081 |
| NADA (GPTQ) | **10.996** | **15.985** | **17.120** | **23.208** | **6.710** | **10.440** | **12.396** | **19.751** |
| MicroRot + NADA (RTN) | 13.056 | 19.722 | 23.123 | 27.743 | 7.129 | 11.228 | 13.085 | 20.918 |
| MicroRot + NADA (GPTQ) | 11.006 | 16.123 | 17.859 | 23.758 | 6.730 | 10.477 | 12.614 | 19.990 |
| SmoothQuant + NADA (RTN) | 16.186 | 23.576 | $2.48\times10^6$ | $2.37\times10^6$ | 19.170 | 27.600 | 67.333 | 94.248 |
| SmoothQuant + NADA (GPTQ) | 11.246 | 16.528 | 17.708 | 23.396 | 6.811 | 10.603 | 12.696 | 20.148 |

NADA can use either RTN or GPTQ as its weight-quantization backbone, while arbitrary preprocessing combinations are not necessarily compatible. The GPTQ-based configuration is generally more accurate because its weight correction compensates for the quantization error remaining after activation preprocessing.

Prepending MicroRot or SmoothQuant does not consistently improve NADA. In particular, SmoothQuant combined with RTN becomes unstable on several models, whereas its GPTQ-based counterpart remains numerically stable. These results show that NADA is modular with respect to the weight-quantization backbone, but the benefits of additional preprocessing transformations are not necessarily additive. The default NADA configuration therefore uses GPTQ without additional MicroRot or SmoothQuant preprocessing.

## M ROBUSTNESS TO RANDOM SEEDS

We evaluate the stability of NADA under five random seeds controlling the calibration-data sampling. The model, calibration size, quantization configuration, and evaluation protocol are otherwise unchanged. Table 17 reports the individual results and their mean and standard deviation.

Table 17: Perplexity of NADA under different random seeds. The final row reports the mean and sample standard deviation over five runs.

| Seed | SmolLM2-135M | | Qwen3-0.6B | | Llama3.2-1B | | Qwen3-1.7B | |
|---|---|---|---|---|---|---|---|---|
| | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 | WKT2 | C4 |
| 0 | 18.841 | 25.266 | 23.146 | 33.428 | 10.996 | 15.985 | 17.120 | 23.208 |
| 42 | 18.770 | 25.255 | 23.400 | 33.488 | 10.948 | 15.981 | 17.752 | 23.537 |
| 50 | 18.916 | 25.406 | 23.225 | 33.001 | 10.943 | 15.956 | 17.518 | 23.177 |
| 100 | 18.744 | 25.325 | 23.573 | 33.261 | 10.970 | 16.187 | 17.142 | 23.366 |
| 163 | 18.853 | 25.307 | 23.405 | 32.997 | 10.963 | 15.965 | 17.684 | 23.457 |
| Mean ± Std. | 18.825 ± 0.069 | 25.312 ± 0.060 | 23.350 ± 0.168 | 33.235 ± 0.231 | 10.964 ± 0.021 | 16.015 ± 0.097 | 17.443 ± 0.298 | 23.349 ± 0.156 |

NADA exhibits limited variation across calibration seeds, with standard deviations below 0.30 on both evaluation corpora for all tested models. The variation is particularly small on SmolLM2-135M and Llama3.2-1B, while Qwen3-1.7B is relatively more sensitive on WikiText-2. Nevertheless, the mean results preserve the improvements observed in the main experiments, showing that NADA's effectiveness is not dependent on a single favorable calibration sample.