# SCOPE: FROM ANALYTICAL SCALE ORACLE TO CANDIDATE SCALE SEARCH FOR NVFP4 QUANTIZED TRAINING

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

## ABSTRACT

Recent hardware support for microscaled FP4 formats has made NVFP4 a compelling target for reducing the cost of large language model pre-training. However, existing NVFP4 training recipes still show a noticeable quality gap to BF16, primarily due to forward weight-activation quantization under the sparse FP4 codebook and shared block scale. We propose **SCOPE**, a forward-centered NVFP4 quantization framework that combines an analytical scale oracle with a deployable candidate search. SCOPE first introduces an analytical per-block scale oracle based on FP4 rounding breakpoints to characterize the reconstruction improvement recoverable by adaptive scale selection. For deployment, SCOPE replaces online breakpoint enumeration with a compact fixed-candidate search, preserving data-dependent scale selection while keeping the online path simple. Backward gradient-to-quantization-noise diagnostics further show that lightweight SR/Had-SR estimators remain well above the theoretically motivated diagnostic reference in our setting, allowing us to use a low-overhead backward path and focus the main optimization effort on forward scale selection. On NanoChat pre-training up to 1.9B parameters and 38B tokens, SCOPE reduces the BF16 BPB gap to $1.01\%$, a relative gap reduction of $\sim 30\%$ over Quartet-II and $\sim 47\%$ over the NVIDIA NVFP4 recipe. In a matched 500M experiment, SCOPE reduces the measured complete-recipe training time by 16.5% relative to Quartet-II while achieving better validation quality. Additional PTQ and pre-training experiments show transfer across dense LLMs, MoE models, and vision-language settings.

## 1 INTRODUCTION

Reducing the precision of general matrix multiplications (GEMMs) is a direct way to reduce the cost of LLM training, motivating the progression from FP16/BF16 (Kalamkar et al., 2019) to FP8/INT8 (Liu et al., 2024a; Ashkboos et al., 2025; Xi et al., 2024) and more recently to 4-bit floating point training (Xi et al., 2023; Abecassis et al., 2025). NVIDIA Blackwell provides native tensor-core support (NVIDIA, 2024; 2023) for both microscaled MXFP8 and the NV-format NVFP4, which pairs E2M1 values with a fine-grained block factor (block size 16, E4M3 factor) for better local dynamic-range adaptation than plain FP4 (Abecassis et al., 2025; Panferov et al., 2026; Cook et al., 2025).

Stable, accurate NVFP4 training nonetheless remains hard. Recent work converges on one principle: forward and backward must not be quantized symmetrically (Abecassis et al., 2025; Dong et al., 2026; Chen et al., 2025b; Panferov et al., 2026; Cook et al., 2025; Wang et al., 2025). The backward pass requires a low-bias gradient estimator (stochastic rounding (Abecassis et al., 2025), or correction-based MS-EDEN (Panferov et al., 2026)). The forward pass requires high-fidelity representation under a sparse FP4 codebook, where one shared block scale must align 16 values with only a few spaced levels. Figure 1(b) makes the asymmetry concrete: under modern backward configurations, gradient-only quantization (G-only, MS-EDEN) tracks BF16 across all $D/N$, while every weight–activation (WA) scheme—RTN, 4over6 (Cook et al., 2025), and ours—accumulates a gap that *grows* with training. *Forward WA quantization, not gradient quantization, is the dominant loss source in NVFP4.*



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.

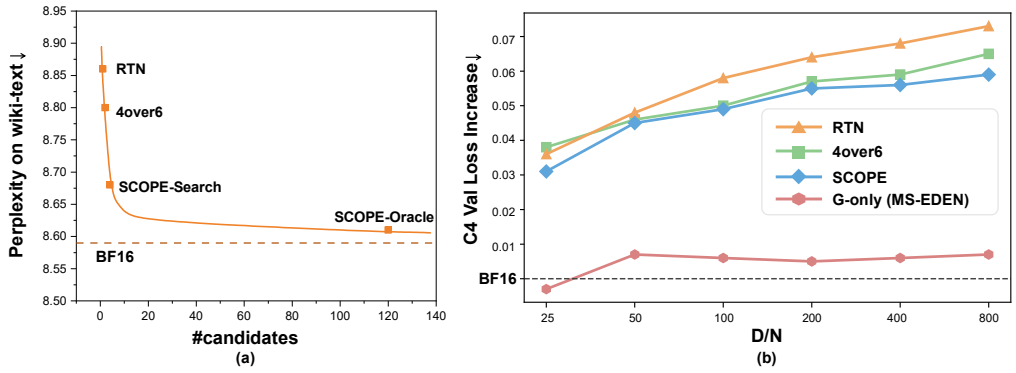


Figure 1: Forward WA quantization is the dominant NVFP4 bottleneck. (a) WikiText PPL vs. per-block candidate count under PTQ; (b) 50M pre-training loss increase over BF16 vs. token-to-parameter ratio $D/N$.

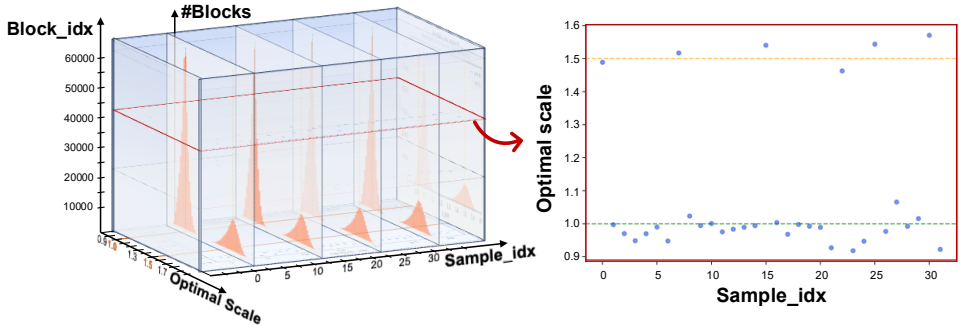


Figure 2: Optimal NVFP4 block scales are stable per tensor (left, bimodal at 1.0 and 1.5) but unpredictable at fixed block indices across samples (right).

Forward representation fidelity therefore reduces to one question: which scale should each NVFP4 block use? Even a small scale change can alter FP4 code assignments at rounding breakpoints, yielding a continuous piecewise-quadratic reconstruction objective with nonsmooth transitions. Static or position-conditioned scales fail: as Figure 2 shows, the per-tensor distribution of optimal block scales is sharply bimodal with stable peaks near 1.0 and 1.5 (left), yet the optimum at any fixed block index jumps unpredictably between these regimes across samples (right). Selection must be data-dependent at the block level, while the candidate set can be derived offline from tensor-level statistics. This raises the central question: *how much can data-dependent per-block scale selection improve NVFP4 forward quantization, and can it be obtained without an expensive online optimizer?*

**SCOPE** (Scale Choice Optimization through Piecewise Error) is proposed to address both. Reconstruction error is a piecewise-quadratic function of the continuous scale with breakpoints at midpoints between adjacent FP4 codewords; we derive a closed-form oracle characterizing the per-block optimum, reported as SCOPE-Oracle in Figure 1(a) and used purely for analysis. For deployment, SCOPE-Search approximates this continuous effective-factor oracle with a small, offline-constructed candidate set—4 scales for weights, 8 for activations ($K_W = 4, K_A = 8$)—and each block online selects the candidate with the lowest reconstruction error. This decouples *where candidates come from* (offline, tensor-level statistics) from *how they are used* (a fixed, online, data-dependent rule); Figure 1(a) shows $K_W = 4, K_A = 8$ already matches most of the benefit of the SCOPE-Oracle.

**Contributions:**

- **Analytical upper bound for NVFP4 scale selection: SCOPE-Oracle.** A piecewise-quadratic analytical scale oracle over FP4 rounding breakpoints that quantifies the forward reconstruction error recoverable by per-block scale choice in NVFP4.
- **Deployable fixed-candidate algorithm: SCOPE-Search.** A practical reformulation of online scale optimization as fixed-candidate ($K_W = 4, K_A = 8$) reconstruction-error mini-

mization over NVFP4-realizable scales, avoiding online breakpoint enumeration, sorting, and dense grid search while preserving data-dependent block-level selection.

- **Diagnostic justification of the forward-centered design.** Gradient-to-quantization-noise diagnostics show lightweight SR/Had-SR remain well above the theoretically motivated diagnostic reference in our setting, supporting investment in the forward path with a low-overhead backward.
- **State-of-the-art NVFP4 pre-training accuracy.** On NanoChat pre-training up to 1.9B parameters and 38B tokens, SCOPE closes the BF16 BPB gap to $1.01\%$—a relative improvement of $\sim 30\%$ over Quartet II and $47\%$ over the NVIDIA NVFP4 recipe.

## 2 Related Work

**Model quantization.** Model quantization (Han et al., 2015) reduces memory footprint and computational cost by representing weights, activations, gradients, or key–value (KV) caches (Hooper et al., 2024; Liu et al., 2024d) with low-precision formats. Post-training quantization (PTQ) quantizes a pretrained model without gradient-based retraining, and prior methods reduce quantization error through weight reconstruction, calibration, smoothing, rotation, clipping, or other transformations (Lin et al., 2024; Frantar et al., 2022; Zhang et al., 2023; Zhou et al., 2025; Li et al., 2026; Xiao et al., 2023; Liu et al., 2024c; Tseng et al., 2024). Quantization-aware training (QAT) instead adapts model parameters under simulated quantization effects (Chen et al., 2025a; Liu et al., 2024b; Xin et al., 2026). In this work, we use *quantized training* to refer to methods that directly employ low-precision operands during optimization, rather than using quantization only to prepare a pretrained model for low-precision inference.

**Quantized training and NVFP4.** Quantized training has progressed toward increasingly low-precision forward and backward computation during pre-training or post-training adaptation (Chen et al., 2025b; Panferov et al., 2026; Wang et al., 2025; Ashkboos et al., 2025; Castro et al., 2025). Recent FP4 training recipes increasingly adopt NVFP4 (Abecassis et al., 2025; Cook et al., 2025; 2026; Chmiel et al., 2025), which represents E2M1 FP4 values together with fine-grained FP8 block factors and a tensor-level FP32 factor. These factors determine the effective quantization step of each block and therefore directly affect forward quantization error. We provide the exact NVFP4 representation and the terminology used throughout this paper in Appendix A.

**Adaptive quantization parameters.** A broad line of work improves low-precision quantization by learning, predicting, or calibrating quantization parameters such as step sizes, clipping ranges, and transformation parameters (Zhou et al., 2025; Liu et al., 2024b; Chen et al., 2025a; Shao et al., 2023; Kryzhanovskiy et al., 2021). SCOPE follows the general principle of adaptive quantization but focuses on *block-wise scale selection* for NVFP4 training. Rather than treating the scale as a fixed or directly learned parameter, SCOPE first characterizes the attainable benefit of adaptive scale selection through an analytical rounding-breakpoint oracle, and then approximates this oracle with a small calibrated candidate set for efficient online block-wise selection.

## 3 SCOPE: Scale Choice Optimization through Piecewise Error

SCOPE is motivated by two key observations in NVFP4 training. First, the forward weight-activation quantization is the dominant NVFP4 bottleneck, where the sparse FP4 codebook and shared block scale directly affect the representations passed to subsequent layers. Second, the preferred scale is highly input-dependent: as shown in Figure 2, even at the same block position, the oracle scale can vary substantially across samples, making static or position-wise scale rules unreliable. These observations motivate a forward-centered scale-selection framework. In this section, we formulate the forward scale-selection problem in Section 3.1, derive an analytical oracle to characterize its upper bound in Section 3.2, replace the oracle with a deployable fixed-candidate search in Section 3.3, and finally justify the lightweight backward path used by SCOPE in Section 3.4.

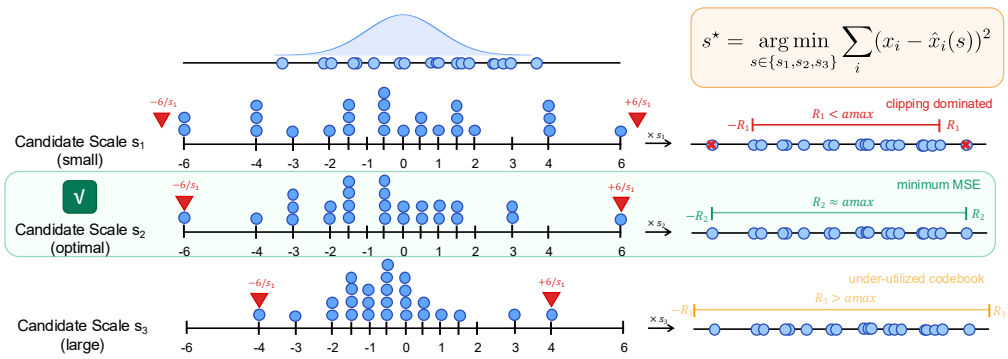


Figure 3: **Overview of our proposed SCOPE.** A small scale leads to over-spreaded values on the FP4 codebook (clipping) while a large scale leads to concentrated values near low-magnitude codes (under-utilization).

### 3.1 Forward Scale Selection in NVFP4

We follow the NVFP4 notation introduced in Section 2. For block $g$, let $\Delta_g$ denote the effective dequantization factor. SCOPE optimizes this effective factor through a scale $s$ relative to the block maximum:

$$M_g = \max_{i\in g} |x_i|, \qquad \Delta_g(s) = \frac{M_g}{max(\mathcal{C})} s. \tag{1}$$

Here, $max(\mathcal{C}) = 6$ is the maximum FP4 code and $s$ is a search scale used by SCOPE rather than an additional stored quantization parameter.

Therefore, given $s$, the block reconstruction error is

$$L_q(s; x_g) = \sum_{i\in g} \left( x_i - \Delta_g(s) Q_{\text{FP4}} \left( \frac{6x_i}{M_g s} \right) \right)^2, \tag{2}$$

This objective exposes why scale selection is important for NVFP4. The scale $s$ controls the trade-off between clipping and codebook utilization: smaller values allocate too little range and may saturate large elements, while larger values avoid clipping but concentrate values near low-magnitude codes and under-utilize the FP4 codebook, as illustrated in Figure 3.

Because the FP4 codebook is sparse and non-uniform, changing $s$ can alter many code assignments within a block. SCOPE therefore treats forward quantization as a scale-selection problem:

$$s_g^\star = \arg\min_{s\in\mathcal{S}} L_q(s; x_g). \tag{3}$$

Different choices of $\mathcal{S}$ correspond to different algorithms. A single default scale recovers RTN-style quantization, dense or continuous search gives an oracle upper bound, and SCOPE-Search uses a compact fixed candidate set.

### 3.2 Analytical Scale Oracle

We first construct an analytical oracle to quantify the upper bound of adaptive scale selection. For one block $g$, we use the reconstruction objective $L_q(s; x_g)$ defined above. The key observation is that the objective is piecewise quadratic in $s$: although the quantized code assignment changes discontinuously with $s$, it changes only when a normalized value crosses an FP4 rounding threshold.

The positive rounding thresholds of the NVFP4 codebook are

$$\mathcal{T} = \{0.25, 0.75, 1.25, 1.75, 2.5, 3.5, 5.0\}.$$

For each element $x_i$ and threshold $\tau \in \mathcal{T}$, the scale value at which the FP4 assignment changes is

$$b_{i,\tau} = \frac{max(\mathcal{C})|x_i|}{M_g \tau} = \frac{6|x_i|}{M_g \tau} \tag{4}$$

**Algorithm 1** Analytical Scale Oracle for One NVFP4 Block

**Require:** Block $x_g \in \mathbb{R}^B$, search range $[s_{\min}, s_{\max}]$
**Require:** FP4 thresholds $\mathcal{T} = \{0.25, 0.75, 1.25, 1.75, 2.5, 3.5, 5.0\}$
**Ensure:** Oracle scale $s^\star$

$M \leftarrow \max_i |x_i|$
**if** $M = 0$ **then**
  **return** $s^\star = 1$
**end if**
$\mathcal{B} \leftarrow \left\{ \frac{6|x_i|}{M_g \tau} : i = 1, \ldots, B,\ \tau \in \mathcal{T} \right\}$ ▷ scales where FP4 codes change
$\mathcal{P} \leftarrow \text{sort}\left(\{s_{\min}, s_{\max}\} \cup (\mathcal{B} \cap (s_{\min}, s_{\max}))\right)$ ▷ partition into fixed-code segments
$best \leftarrow +\infty, \quad s^\star \leftarrow s_{\min}$
**for** each consecutive interval $[s_l, s_r]$ in $\mathcal{P}$ **do**
  $s_m \leftarrow (s_l + s_r)/2$
  $v_i \leftarrow Q_{\mathcal{C}_4}\left(6x_i/(M s_m)\right)$ ▷ code assignment inside this segment
  $u^\star \leftarrow \frac{\sum_i v_i x_i}{\sum_i v_i^2}$
  $s_{\text{seg}} \leftarrow \text{clip}\left(6u^\star/M,\ s_l,\ s_r\right)$ ▷ closed-form optimum clipped to segment
  $\hat{x}_i \leftarrow \frac{M s_{\text{seg}}}{6} Q_{\mathcal{C}_4}\left(6x_i/(M s_{\text{seg}})\right)$ ▷ re-quantize for actual error
  $err \leftarrow \sum_i (x_i - \hat{x}_i)^2$
  **if** $err < best$ **then**
    $best \leftarrow err, \quad s^\star \leftarrow s_{\text{seg}}$
  **end if**
**end for**
**return** $s^\star$

These breakpoints partition $[s_{\min}, s_{\max}]$ into segments within which the FP4 code assignment is fixed. Therefore, instead of searching over all possible scales directly, the oracle only needs to solve one scalar least-squares problem per segment. Algorithm 1 summarizes the resulting search. For a segment $[s_l, s_r]$, we evaluate its midpoint $s_m$ to obtain the fixed codes $v_i = Q_{\mathcal{C}_4}(6x_i/(M_g s_m))$. With $u = M_g s/6$, the segment-wise objective reduces to the one-dimensional least-squares problem $\min_u \sum_i (x_i - u v_i)^2$, yielding

$$u^\star = \frac{\sum_i v_i x_i}{\sum_i v_i^2}, \qquad s^\star_{\text{seg}} = \text{clip}\left(\frac{6u^\star}{M_g}, s_l, s_r\right). \tag{5}$$

For each segment, the oracle computes the clipped closed-form solution $s^\star_{\text{seg}}$, re-evaluates the block reconstruction error at this scale, and finally returns the scale that achieves the minimum error across all segments. The complete derivation is provided in Appendix E.

### 3.3 From Analytical Scale Oracle to Candidate Scale Search

SCOPE-Oracle characterizes the best scale choice under a relaxed continuous effective-factor assumption. This relaxation removes the secondary discretization introduced by the E4M3 block-factor lattice, whose error is much smaller than that from assigning values to the sparse and non-uniform FP4 codebook (Cook et al., 2025). The normalized SCOPE scale also provides a more compact search domain than the raw FP8 factor, which additionally absorbs variation from the block maximum, tensor factor, and E4M3 lattice.

However, the oracle is not suitable as an online training quantizer. It requires generating FP4 rounding breakpoints, sorting them, and solving one segment-wise least-squares problem for every quantization block. We therefore replace this oracle with a fixed-candidate search as the deployment approximation.

Concretely, as shown in Figure 3, SCOPE-Search uses a compact candidate set $\mathcal{S}$ of normalized SCOPE scales:

$$s_{\text{SCOPE}}(x_g) = \arg\min_{s \in \mathcal{S}} L_q(s; x_g) \tag{6}$$

Table 1: End-to-end NanoChat comparison of backward estimators. Mean ± standard deviation is reported for validation BPB.

| Metric | BF16 | SCOPE + MS-EDEN | SCOPE + Had-SR |
|---|---|---|---|
| Val BPB ↓ | $0.7689 \pm 0.0011$ | $0.7771 \pm 0.0017$ | $\mathbf{0.7763} \pm 0.0014$ |
| Increase over BF16 ↓ | – | 1.07% | **0.96%** |

The candidate set $\mathcal{S}$ is obtained once from calibration and is then kept fixed in the experiments below. During calibration, we collect representative forward blocks, evaluate reconstruction quality over a range of candidate scales, and select a small set that provides a good quality–overhead trade-off.

We calibrate the candidate set by directly optimizing the deployed reconstruction objective. For tensor type $T \in \{W, A\}$, let $\mathcal{D}_T$ be the calibration blocks and let $\mathcal{U}$ be a dense pool of candidate scales guided by the oracle-scale distribution. For a target budget $K_T$, we select

$$\mathcal{S}_T^\star = \arg \min_{\mathcal{S} \subset \mathcal{U},\, |\mathcal{S}| = K_T} \frac{1}{|\mathcal{D}_T|} \sum_{x_g \in \mathcal{D}_T} \min_{s \in \mathcal{S}} L_q(s; x_g) \tag{7}$$

We optimize this objective greedily by adding the scale that gives the largest reduction in calibration reconstruction error at each step.

We use $K_W = 4, K_A = 8$ as deployment-friendly budgets. This asymmetric budget reflects the more stationary statistics of weights and the stronger sample dependence of activations. The resulting $K_W = 4, K_A = 8$ configuration is fixed across all model families, datasets, and training runs, and is not tuned on downstream tasks or final validation metrics. Appendix G additionally studies an explicitly labeled vision-specific recalibration as an optional domain adaptation. Exact values and calibration details are given in Appendix B.

At runtime, SCOPE-Search replaces the oracle's breakpoint enumeration and sorting with reconstruction-error evaluation over the fixed offline-calibrated candidates, while preserving data-dependent block-level scale selection.

### 3.4 Backward Pass Quantization Path

SCOPE focuses on forward scale selection, but full NVFP4 training also requires quantized backward GEMMs. Stochastic rounding (SR) (Croci et al., 2022) is unbiased for normalized values before clipping, but introduces rounding noise on the coarse FP4 lattice. MS-EDEN (Panferov et al., 2026) reduces local backward quantization error through correction-based group-wise estimation, at additional quantization overhead. We therefore assess whether this additional accuracy is necessary under our training budget, rather than minimizing backward error in isolation.

Using backward GEMM outputs as gradient proxies, we measure the gradient-to-quantization-noise ratio (GNR): the BF16 result norm divided by the quantization-error norm. On a 500M LLaMA-family model, SR and its Hadamard-assisted variant (Had-SR) achieve GNRs of 6.77–9.51 (Table 2), well above the diagnostic reference $\sqrt{3}$ (Chmiel et al., 2025). This assumption-dependent reference is not a universal convergence guarantee. Although MS-EDEN yields lower local error, the SR-based estimators already operate in a signal-dominant regime and have faster standalone quantization kernels in our implementation.

We further replace Had-SR with MS-EDEN throughout NanoChat training, keeping the forward quantizer and other settings unchanged across three matched seeds. At the fixed 11B-token endpoint, the lower local error does not translate into an improvement in mean validation BPB (Table 1). Together, these results support a lightweight SR/Had-SR backward path under the evaluated training budgets, allowing us to focus the main optimization effort on forward representation fidelity. The full recipe, estimator definitions, and diagnostic analysis are provided in Appendix D, with additional SR variants in Appendix D.2.

Table 2: Backward GEMM diagnostics on a 500M LLaMA-family model.

*Note:* Kernel speedup is measured for the standalone backward quantization kernel on a $4096 \times 8192$ matrix. We normalize to MS-EDEN to highlight the relative overhead of the correction-based estimator.

| GEMM | Method | Rel-MSE ↓ | GNR ↑ | Margin over $\sqrt{3}$ | Kernel Speedup ↑ |
|---|---|---|---|---|---|
| $\nabla X = EW$ | MS-EDEN | 0.01539 | 8.06 | $4.7\times$ | $1.0\times$ |
| | SR | 0.02181 | 6.77 | $3.9\times$ | $1.5\times$ |
| | Had-SR-16 | 0.02128 | 6.86 | $4.0\times$ | $1.1\times$ |
| $\nabla W = E^\top X$ | MS-EDEN | 0.01043 | 9.79 | $5.7\times$ | $1.0\times$ |
| | SR | 0.01105 | 9.51 | $5.5\times$ | $1.5\times$ |
| | Had-SR-16 | 0.01399 | 8.45 | $4.9\times$ | $1.1\times$ |

Table 3: PTQ perplexity on WikiText-2 across multiple models.

| | Llama 3 Token PPL (↓) | | | Qwen 3 Token PPL (↓) | | |
|---|---|---|---|---|---|---|
| Method | 1B | 8B | 70B | 1.7B | 8B | 32B |
| BF16 | 8.59 | 5.55 | 2.47 | 14.95 | 8.59 | 6.70 |
| RTN | 10.16 | 6.16 | 3.24 | **16.29** | 8.86 | 7.02 |
| + 4over6 | 9.93 | 6.09 | 3.12 | 16.65 | 8.80 | 6.97 |
| + SCOPE-Search | **9.87** | **6.06** | **2.79** | 16.56 | **8.68** | **6.86** |
| + *SCOPE-Oracle* | 9.21 | 5.81 | 2.76 | 16.57 | 8.61 | 6.83 |

# 4 Experiments

We evaluate SCOPE in three settings: (i) post-training quantization (PTQ) to isolate forward scale selection on pretrained models, (ii) LLaMA-family pre-training on C4 (Raffel et al., 2020) to test transfer to optimization dynamics, and (iii) NanoChat pre-training as an independent pipeline with different architecture and recipe. SCOPE uses a fixed candidate set calibrated once on a small C4 subset and kept unchanged thereafter. All NVFP4 methods share the same FP4 codebook and block size, differing only in forward scale selection and, for full training, the backward quantization path.

## 4.1 Post-Training Quantization Evaluation

PTQ keeps weights fixed, removing optimizer-dynamics confounders and isolating forward optimization. We quantize forward weights and activations with RTN, 4over6 (Cook et al., 2025), and two SCOPE variants: **SCOPE-Oracle** and **SCOPE-Search**. In subsequent pre-training experiments, SCOPE refers to the full recipe with fixed-candidate search.

Table 3 reports WikiText-2 (Merity et al., 2016) perplexity. SCOPE adapts the scale per block via direct reconstruction-error comparison and replaces 4over6's fixed heuristic alternatives with a compact offline-selected candidate set, yielding modest but consistent gains across model families (Dubey et al., 2024; Yang et al., 2025) and datasets except Qwen3-1.7B. Table 4 confirms the benefit transfers beyond perplexity: on ARC, HellaSwag and BoolQ (Clark et al., 2018; Zellers et al., 2019; Clark et al., 2019), SCOPE consistently improves the average over RTN and 4over6.

## 4.2 LLaMA-Family Quantized Pre-Training

We test whether SCOPE transfers from PTQ to quantized pre-training on C4, sharing the LLaMA-family architecture, optimizer, data, and schedule with the BF16 baseline; we report validation-loss increase over BF16.

**Forward W/A ablation.** On a 50M model with a 1:800 parameter-to-token ratio (Figure 4(a)), every NVFP4 forward quantizer leaves a persistent BF16 gap, and SCOPE achieves the smallest among RTN, 4over6, and itself, confirming forward W/A as a major degradation source.

Table 4: Downstream PTQ results on lm_eval benchmarks. We report accuracy (%) on BoolQ, ARC-Easy, ARC-Challenge, and HellaSwag across Llama-3.2-1B and 3.1-8B models.

| | BoolQ | | ARC-E | | ARC-C | | HellaSwag | | Avg. | |
|---|---|---|---|---|---|---|---|---|---|---|
| Method | 1B | 8B | 1B | 8B | 1B | 8B | 1B | 8B | 1B | 8B |
| BF16 | 63.85 | 83.00 | 66.29 | 82.32 | 31.48 | 52.47 | 48.04 | 60.78 | 52.42 | 69.64 |
| RTN | 61.25 | 81.31 | 63.64 | 81.44 | 30.55 | 50.51 | 46.49 | 60.05 | 50.48 | 68.33 |
| + 4over6 | 61.38 | 81.28 | 63.93 | 81.65 | 31.23 | **51.54** | 46.43 | 60.01 | 50.74 | 68.62 |
| + SCOPE-Search | **62.78** | **81.93** | **64.77** | **81.99** | **32.25** | 50.77 | **47.01** | **60.54** | **51.70** | **68.81** |
| + *SCOPE-Oracle* | 64.31 | 81.35 | 65.45 | 82.32 | 32.94 | 50.68 | 46.77 | 60.27 | 52.37 | 68.66 |

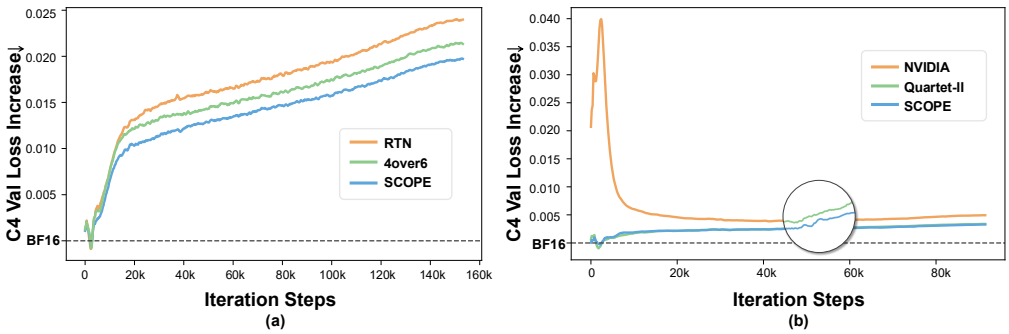


Figure 4: **LLaMA-family quantized pre-training on C4.** (a) 50M forward WA ablation. (b) 500M full W/A/G setting. SCOPE consistently achieves the smallest validation-loss gap to BF16.

**Full W/A/G training.** On a 500M model with a 1:50 ratio (Figure 4(b)), where weights, activations, and backward gradient GEMMs are all quantized, SCOPE remains stable and achieves a smaller BF16 gap than both the NVIDIA NVFP4 recipe and Quartet-II; the advantage persists in the late-stage trend, indicating the gain is not an early-training transient.

### 4.3 NanoChat Pre-Training

Following Quartet-II (Panferov et al., 2026), we evaluate SCOPE in the NanoChat pipeline (Karpathy, 2025), which differs in optimizer, normalization, activations, data mixture, and schedule, providing an independent testbed. We compare BF16, the NVIDIA NVFP4 recipe (Abecassis et al., 2025), Quartet-II, and SCOPE under identical NanoChat hyperparameters; all quantized methods use full W/A/G in pre-training. Since SCOPE targets pre-training quality, we treat final validation BPB as the primary metric.

Table 5 and 6 show that on Speedrun (560M, 11B tokens) SCOPE reduces the BF16 BPB gap to 0.96%, versus 1.69% (NVIDIA), 2.43% (4over6) and 1.38% (Quartet-II); on the 1.9B/38B setting it reaches 1.01%, a relative gap reduction of $\sim$47% over NVIDIA and $\sim$30% over Quartet-II. Detailed task-level accuracy results are provided in Table 13.

### 4.4 End-to-End Efficiency and Scaling Behavior

We measure end-to-end wall-clock time for the complete training recipes on the 500M LLaMA model (1:50 parameter-to-token ratio, 4$\times$ B30 GPUs) under identical settings. NVIDIA NVFP4, Quartet-II, and SCOPE take 950, 1283, and 1071 min, respectively. Thus, SCOPE reduces the measured training time by 16.5% relative to Quartet-II (1071 versus 1283 min), but is 12.7% slower than the NVIDIA NVFP4 reference recipe, while achieving the best validation quality among the evaluated quantized methods. These timings cover the complete training recipe, including both forward and backward paths.

Table 5: NanoChat Speedrun results with 560M parameters and 11B training tokens. Mean ± standard deviation is reported for validation BPB. Accuracy is averaged over the full evaluation suite. Bold indicates the best result among quantized methods.

| Metric | BF16 | 4over6 | NVIDIA | Quartet-II | SCOPE |
|---|---|---|---|---|---|
| Val BPB ↓ | 0.7689 | 0.7876 | 0.7819 | 0.7795 | **0.7763** |
| Std. | 0.0011 | 0.0033 | 0.0017 | 0.0012 | 0.0014 |
| Increase over BF16 (%) ↓ | – | 2.43 | 1.69 | 1.38 | **0.96** |
| Avg. Acc (%) ↑ | 38.57 | 36.52 | 37.89 | 37.50 | **38.23** |

Table 6: NanoChat $1000 pre-training results with 1.9B parameters and 38B training tokens. Bold indicates the best result among quantized methods.

| Metric | BF16 | NVIDIA | Quartet-II | SCOPE |
|---|---|---|---|---|
| Val BPB ↓ | 0.6925 | 0.7058 | 0.7025 | **0.6995** |
| Increase over BF16 (%) ↓ | – | 1.92 | 1.44 | **1.01** |

**Scaling of cost and benefit.** At fixed candidate budgets, larger hidden width and more tokens per forward pass increasingly amortize activation- and weight-side scale search, respectively. On GB200, the additional forward-path overhead is approximately 7% for a LLaMA-3.1-70B-style configuration and below 5% for a 175B-style configuration with $(K_W, K_A) = (4, 8)$. Meanwhile, in our controlled forward W/A ablation, SCOPE's validation-loss advantage over RTN grows with the training-token budget. These complementary trends support a favorable quality–overhead trade-off in the evaluated configurations as model width, per-forward token count, and training budget increase; they do not constitute a general system-level speedup guarantee. Detailed measurements and scaling analysis are provided in Appendix H.

## 5 Limitations

SCOPE still leaves a residual gap to lossless NVFP4 forward quantization and our conclusions are still limited to the evaluated training scales and shared hyperparameter settings rather than method-specific tuning or frontier-scale training. Our current implementation also uses a research-oriented quantization path instead of a fully optimized production FP4 stack, so the reported efficiency does not represent the maximum attainable system-level speedup or establish a general speedup over BF16. Moreover, while SR/Had-SR is sufficient under the evaluated training budgets and settings, lower-noise regimes may benefit from stronger backward estimators. We provide a more detailed discussion in Appendix C.

## 6 Conclusion

We propose SCOPE, a forward-centered NVFP4 training framework that combines an analytical scale oracle for upper-bound analysis with a deployable candidate scale search. We further analyze backward pass quantization and show that, under the experimental settings, lightweight SR/Had-SR estimators are sufficient to support a low-overhead backward path. Across PTQ and quantized pre-training experiments, SCOPE consistently improves NVFP4 quality and achieves the best validation performance among the compared quantized methods with moderate overhead.

### AI use statement

We used generative AI tools to assist with parts of the PyTorch and CUDA implementation, provide limited feedback on experimental design and configuration, assist with literature discovery, and improve the clarity and readability of the manuscript. We also used these tools for language polishing and translation.

All AI-assisted code was reviewed and tested by the authors. Suggested experimental configurations were independently validated through experiments or checked against original papers and official documentation, and suggested literature was verified against the original sources. Generative AI was not used to generate or modify the reported experimental results. The authors take full responsibility for the final content, claims, code, and experimental results presented in this work.

### Ethics statement

This work studies efficient low-precision training of large language models and does not involve human subjects or the collection of personal or sensitive data. Our experiments use publicly available models and datasets under their respective licenses and terms of use. We do not release new datasets containing user-generated or personally identifiable information.

### Reproducibility statement

We provide the formulation and implementation details of SCOPE in the main text and appendix, including the exact candidate sets and calibration protocol, the NVFP4 forward and backward training recipe, stochastic-rounding variants, and the complete derivation of the analytical scale oracle. The experimental sections document the evaluated models, datasets, training configurations, evaluation protocols, hardware settings, and timing methodology. Additional ablations, scale-dynamics analyses, generalization experiments, and component-level efficiency and scaling measurements are provided in the appendix. Together, these details are intended to facilitate reproduction of both the training-quality and efficiency results.

## References

Felix Abecassis, Anjulie Agrusa, Dong Ahn, Jonah Alben, Stefania Alborghetti, Michael Andersch, Sivakumar Arayandi, Alexis Bjorlin, Aaron Blakeman, Evan Briones, et al. Pretraining large language models with nvfp4. *arXiv preprint arXiv:2509.25149*, 2025.

Saleh Ashkboos, Mahdi Nikdan, Soroush Tabesh, Roberto L Castro, Torsten Hoefler, and Dan Alistarh. Halo: Hadamard-assisted lower-precision optimization for llms. *arXiv preprint arXiv:2501.02625*, 2025.

Roberto L Castro, Andrei Panferov, Soroush Tabesh, Oliver Sieberling, Jiale Chen, Mahdi Nikdan, Saleh Ashkboos, and Dan Alistarh. Quartet: Native fp4 training can be optimal for large language models. *arXiv preprint arXiv:2505.14669*, 2025.

Mengzhao Chen, Wenqi Shao, Peng Xu, Jiahao Wang, Peng Gao, Kaipeng Zhang, and Ping Luo. Efficientqat: Efficient quantization-aware training for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10081–10100, 2025a.

Yuxiang Chen, Xiaoming Xu, Pengle Zhang, Michael Beyer, Martin Rapp, Jun Zhu, and Jianfei Chen. Tetrajet-v2: Accurate nvfp4 training for large language models with oscillation suppression and outlier control. *arXiv preprint arXiv:2510.27527*, 2025b.

Brian Chmiel, Maxim Fishman, Ron Banner, and Daniel Soudry. Fp4 all the way: Fully quantized training of llms. *arXiv preprint arXiv:2505.19115*, 2025.

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, 2019.

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

Jack Cook, Junxian Guo, Guangxuan Xiao, Yujun Lin, and Song Han. Four over six: More accurate nvfp4 quantization with adaptive block scaling. *arXiv preprint arXiv:2512.02010*, 2025.

Jack Cook, Hyemin S Lee, Kathryn Le, Junxian Guo, Giovanni Traverso, Anantha P Chandrakasan, and Song Han. Adaptive block-scaled data types. *arXiv preprint arXiv:2603.28765*, 2026.

Matteo Croci, Massimiliano Fasi, Nicholas J Higham, Theo Mary, and Mantas Mikaitis. Stochastic rounding: implementation, error analysis and applications. *Royal Society Open Science*, 9(3), 2022.

Peijie Dong, Ruibo Fan, Yuechen Tao, Di Mou, Wenhu Hu, Zhenheng Tang, Yinghao Yu, Jiamang Wang, Wenbo Su, Guodong Yang, et al. Dissecting outlier dynamics in llm nvfp4 pretraining. *arXiv preprint arXiv:2602.02047*, 2026.

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

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W Mahoney, Yakun S Shao, Kurt Keutzer, and Amir Gholami. Kvquant: Towards 10 million context length llm inference with kv cache quantization. *Advances in Neural Information Processing Systems*, 37:1270–1303, 2024.

Dhiraj Kalamkar, Dheevatsa Mudigere, Naveen Mellempudi, Dipankar Das, Kunal Banerjee, Sasikanth Avancha, Dharma Teja Vooturi, Nataraj Jammalamadaka, Jianyu Huang, Hector Yuen, et al. A study of bfloat16 for deep learning training. *arXiv preprint arXiv:1905.12322*, 2019.

Andrej Karpathy. nanochat: The best chatgpt that $100 can buy, 2025. URL `https://github.com/karpathy/nanochat`.

Vladimir Kryzhanovskiy, Gleb Balitskiy, Nikolay Kozyrskiy, and Aleksandr Zuruev. Qpp: Real-time quantization parameter prediction for deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10692, 2021.

Hanglin Li, Shuchang Tian, Chen Lin, Zhiyong Zhao, and Kun Zhan. Faar: Format-aware adaptive rounding for nvfp4. *arXiv preprint arXiv:2603.22370*, 2026.

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. *Proceedings of machine learning and systems*, 6: 87–100, 2024.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024a.

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, 2024b.

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. *arXiv preprint arXiv:2405.16406*, 2024c.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*, 2024d.

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

NVIDIA. Transformer engine. https://github.com/NVIDIA/TransformerEngine, 2023. GitHub repository, accessed April 13, 2026.

NVIDIA. Nvidia blackwell architecture technical brief, 2024. URL https://resources.nvidia.com/en-us-blackwell-architecture. Technical brief, accessed 2026-04-13.

Andrei Panferov, Erik Schultheis, Soroush Tabesh, and Dan Alistarh. Quartet ii: Accurate llm pre-training in nvfp4 by improved unbiased gradient estimation. *arXiv preprint arXiv:2601.22813*, 2026.

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.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. Omniquant: Omnidirectionally calibrated quantization for large language models. *arXiv preprint arXiv:2308.13137*, 2023.

Albert Tseng, Qingyao Sun, David Hou, and Christopher De. Qtip: Quantization with trellises and incoherence processing. *Advances in Neural Information Processing Systems*, 37:59597–59620, 2024.

Ruizhe Wang, Yeyun Gong, Xiao Liu, Guoshuai Zhao, Ziyue Yang, Baining Guo, Zhengjun Zha, and Peng Cheng. Optimizing large language model training using fp4 quantization. *arXiv preprint arXiv:2501.17116*, 2025.

Haocheng Xi, Changhao Li, Jianfei Chen, and Jun Zhu. Training transformers with 4-bit integers. *Advances in Neural Information Processing Systems*, 36:49146–49168, 2023.

Haocheng Xi, Yuxiang Chen, Kang Zhao, Kai Jun Teh, Jianfei Chen, and Jun Zhu. Jetfire: Efficient and accurate transformer pretraining with int8 data flow and per-block quantization. *arXiv preprint arXiv:2403.12422*, 2024.

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

Meng Xin, Sweta Priyadarshi, Jingyu Xin, Bilal Kartal, Aditya Vavre, Asma Kuriparambil Thekkumpate, Zijia Chen, Ameya Sunil Mahabaleshwarkar, Ido Shahaf, Akhiad Bercovich, et al. Quantization-aware distillation for nvfp4 inference accuracy recovery. *arXiv preprint arXiv:2601.20088*, 2026.

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

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 4791–4800, 2019.

Yijia Zhang, Sicheng Zhang, Shijie Cao, Dayou Du, Jianyu Wei, Ting Cao, and Ningyi Xu. Afpq: Asymmetric floating point quantization for llms. *arXiv preprint arXiv:2311.01792*, 2023.

Zhaojing Zhou, Xunchao Li, Minghao Li, Handi Zhang, Haoshuang Wang, Wenbin Chang, Yiqun Liu, Qingqing Dang, Dianhai Yu, Yanjun Ma, et al. Ccq: Convolutional code for extreme low-bit quantization in llms. *arXiv preprint arXiv:2507.07145*, 2025.

# A NVFP4 Quantization and Terminology

**NVFP4 representation.** NVFP4 represents tensor values using the E2M1 FP4 format together with two levels of scaling factors: a fine-grained FP8 block factor and a tensor-level FP32 factor (Abecassis et al., 2025; Cook et al., 2025; 2026; Chmiel et al., 2025). For a vector $x \in \mathbb{R}^d$, we denote its NVFP4 representation as

$$Q_{\mathrm{NVFP4}}(x) \to \left\{ x^{\mathrm{FP4}} \in \mathcal{C}_4^d,\ x^{\mathrm{FP8}} \in \mathbb{R}^{d/16},\ x^{\mathrm{FP32}} \in \mathbb{R} \right\}, \tag{8}$$

where the normalized E2M1 FP4 codebook is

$$\mathcal{C}_4 = \{0, \pm 0.5, \pm 1, \pm 1.5, \pm 2, \pm 3, \pm 4, \pm 6\}. \tag{9}$$

Each FP8 factor is shared by a block of 16 FP4 values. For element $i$ belonging to block $g = \lfloor i/16 \rfloor$, the corresponding dequantized value is

$$\hat{x}_i = x_i^{\mathrm{FP4}} \cdot x_g^{\mathrm{FP8}} \cdot x^{\mathrm{FP32}}. \tag{10}$$

We therefore define the effective dequantization factor of block $g$ as

$$\Delta_g = x_g^{\mathrm{FP8}} x^{\mathrm{FP32}}. \tag{11}$$

Equation 10 can equivalently be written as

$$\hat{x}_i = x_i^{\mathrm{FP4}} \Delta_g. \tag{12}$$

**Terminology.** NVFP4 literature commonly refers to the FP8 and FP32 quantities above as scales. In this paper, however, we reserve the term ***scale*** for the SCOPE search variable $s$, which is the adaptive quantity studied in our method. To avoid ambiguity, we refer to $x_g^{\mathrm{FP8}}$ and $x^{\mathrm{FP32}}$ as the **block factor** and **tensor factor**, respectively, and to their product $\Delta_g = x_g^{\mathrm{FP8}} x^{\mathrm{FP32}}$ as the **effective dequantization factor**. We follow this terminology consistently throughout the paper.

# B Candidate Sets and Calibration Protocol

SCOPE uses fixed candidate scale sets throughout all experiments. The weight candidate set is

$$\mathcal{S}_W = \{0.9,\ 1.0,\ 1.4,\ 1.5\},$$

and the activation candidate set is

$$\mathcal{S}_A = \{0.9,\ 1.0,\ 1.1,\ 1.3,\ 1.4,\ 1.5,\ 1.6,\ 1.9\}.$$

These candidates are obtained from a one-time calibration on a small C4 subset and then kept fixed across all model families, evaluation datasets, and pre-training runs. We do not tune the candidate sets on downstream tasks or final validation metrics.

**Candidate-budget ablation in quantized training.** The calibration results above motivate $(K_W, K_A) = (4, 8)$ as our default setting and use $K_W = 4, K_A = 4$ when overhead limits. We further examine whether the larger candidate budget translates into an end-to-end benefit during quantized pre-training. Figure 1(b) compares SCOPE with the two-candidate 4over6 setting on a 50M weight–activation quantized pre-training experiment. We additionally perform the ablation at the 560M NanoChat Speedrun scale. To isolate the effect of forward candidate selection, both quantized methods use MS-EDEN for the backward path. As shown in Table 8, SCOPE achieves a lower validation BPB than the smaller-candidate setting, reducing the mean BF16 gap from 1.38% to 1.07%. Together with the 50M result in Figure 4(b), this provides complementary training-level evidence for using the larger candidate set in our default configuration.

To isolate candidate construction from candidate count, we compare three candidate-generation strategies under the same nominal budget $K = 4$: the Greedy/Frequency-TopK set, an E4M3-offset set centered around $1.0$, and a uniform set. All methods use the same native NVFP4 quantization pipeline and are evaluated using per-block reconstruction SSE. Across the seven evaluated projections, Greedy/Frequency-TopK consistently achieves the lowest reconstruction error on both calibration and held-out inputs. Relative to RTN, it reduces the mean SSE by $23.0\%$–$23.1\%$ for weights and by $19.8\%$–$19.9\%$ for activations, whereas the two simple alternatives provide smaller improvements. It also recovers approximately $86\%$ of the reconstruction improvement available from the dense native-grid reference. These results provide reconstruction-level evidence that candidate construction contributes beyond merely increasing the candidate count.

Table 7: Candidate-set calibration on LLaMA-3.1-8B with reported held-out error($1e^{-3}$), oracle recovery and perplexity (PPL).

| $K_A$ | **Selected activation scales** | **Error ↓** | **Recovery ↑** | **PPL ↓** |
|---|---|---|---|---|
| 1 | $\{1.00\}$ | 8.31 | 0 | 6.1536 |
| 2 | $\{1.00, 1.50\}$ | 7.47 | 0.5697 | 6.1164 |
| 4 | $\{1.00, 1.50, 0.90, 1.40\}$ | 7.08 | 0.8305 | 6.1072 |
| 8 | $\{1.00, 1.50, 0.90, 1.40, 1.10, 1.60, 1.30, 1.90\}$ | 6.96 | 0.9120 | 6.0995 |
| 16 | $\{0.80 - 1.70, 1.90, 2.10, 2.30, 2.50, 2.70, 2.90\}$ | 6.96 | 0.9129 | 6.0989 |
| $K_W$ | **Selected weight scales** | **Error ↓** | **Recovery ↑** | **PPL ↓** |
| 1 | $\{1.00\}$ | 8.99 | 0 | 6.1524 |
| 2 | $\{1.00, 1.50\}$ | 7.55 | 0.6041 | 6.1177 |
| 4 | $\{1.00, 1.50, 0.90, 1.40\}$ | 6.93 | 0.8635 | 6.1116 |
| 6 | $\{1.00, 1.50, 0.90, 1.40, 1.60, 1.10\}$ | 6.76 | 0.9365 | 6.1101 |
| 8 | $\{1.00, 1.50, 0.90, 1.40, 1.60, 1.10, 1.30, 1.90\}$ | 6.73 | 0.9482 | 6.1069 |

Table 8: Candidate-budget ablation in the 560M NanoChat Speedrun setting. Both quantized methods use MS-EDEN for the backward path. Mean ± standard deviation is reported over three runs.

| **Metric** | **BF16** | **Quartet-II (4over6)** | **SCOPE** |
|---|---|---|---|
| Val BPB ↓ | $0.7689 \pm 0.0011$ | $0.7795 \pm 0.0012$ | $\mathbf{0.7771} \pm 0.0017$ |
| Increase over BF16 ↓ | – | 1.38% | **1.07%** |

## C LIMITATIONS

**Remaining forward quantization gap.** Although SCOPE reduces forward quantization error compared with existing low-overhead baselines, it does not make NVFP4 forward quantization lossless. The remaining gap still need dedicate design with relatively small overhead.

**Experimental scope and hyperparameters.** For fairness, we use the same training hyperparameters across quantization schemes and do not tune learning rates or schedules for each method. Our largest pre-training experiments are also still smaller than frontier industrial LLM training runs, so further validation leaves for future work.

**System integration.** Our current implementation follows a research-code quantization path rather than a fully integrated Transformer-Engine-style training stack, which limits its effectiveness on truly speedup. As a result, it does not yet realize the production-level speedup possible with fully optimized FP4 kernels.

**Backward estimator regimes.** Our diagnostics show that SR/Had-SR is sufficient under the evaluated training budgets. However, lower-noise regimes, such as longer training, larger effective batches, or late-stage refinement, may require stronger correction-based estimators.

## D PROPOSED NVFP4 TRAINING RECIPE

This section demonstrates the details of our proposed NVFP4 training recipe as shown in Figure 5.

### D.1 BACKWARD ESTIMATORS AND DIAGNOSTICS

**Stochastic rounding and correction-based estimation.** SCOPE focuses on forward scale selection, but full NVFP4 training also requires quantized backward GEMMs. A standard stochastic-rounding (SR) estimator Croci et al. (2022) is attractive because it provides low-bias quantization for gradient-

Table 9: Equal-budget candidate-construction ablation under the native NVFP4 per-block reconstruction objective. All candidate methods use the same nominal budget $K = 4$. Weight ($10^{-5}$) and activation ($10^{-4}$) SSE is measured.

| Operand | Split | RTN | Greedy/Freq. | E4M3-offset | Uniform | Dense Oracle |
|---|---|---|---|---|---|---|
| Weight | Calib- | 9.7414 | **7.4980** | 8.6051 | 8.5305 | 7.1471 |
| Weight | Held | 9.6027 | **7.3874** | 8.4725 | 8.3842 | 7.0365 |
| Activation | Calib- | 2.6923 | **2.1601** | 2.4299 | 2.5779 | 2.0671 |
| Activation | Held | 2.6961 | **2.1609** | 2.4372 | 2.5877 | 2.0656 |

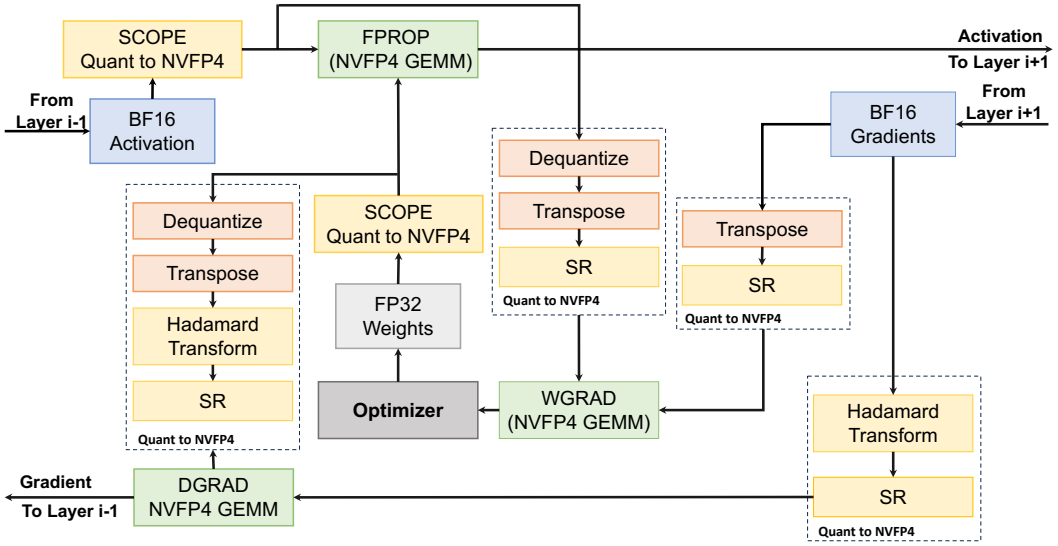


Figure 5: SCOPE NVFP4 training recipes. SCOPE is used in forward pass quantization and stochastic rounding is used in backward pass for unbiased gradient estimation.

related operands. For a normalized scalar $x'$ between two adjacent FP4 levels $c_j \le x' \le c_{j+1}$, stochastic rounding is defined as

$$Q_{\mathrm{SR}}(x') = \begin{cases} c_j, & \text{with probability } \dfrac{c_{j+1} - x'}{c_{j+1} - c_j}, \\ c_{j+1}, & \text{with probability } \dfrac{x' - c_j}{c_{j+1} - c_j}. \end{cases} \tag{13}$$

Thus, $\mathbb{E}[Q_{\mathrm{SR}}(x')] = x'$ before clipping, but the rounding noise is injected on the coarse FP4 lattice. We denote the Hadamard-assisted variant of SR by Had-SR.

MS-EDEN Panferov et al. (2026) further reduces variance through correction-based group-wise estimation, effectively moving much of the stochasticity from FP4 element rounding to the finer FP8-factor level, but at additional overhead.

**Gradient-to-quantization-noise diagnostics.** To decide whether a lower-variance estimator is necessary in our setting, we use backward GEMM outputs as gradient proxies and measure the gradient-to-quantization-noise ratio (GNR) of each backward GEMM. Let $G_{\mathrm{BF16}}$ be the BF16 backward GEMM result and $G_Q$ the corresponding quantized result. We define

$$\mathrm{GNR} = \frac{\|G_{\mathrm{BF16}}\|}{\|G_Q - G_{\mathrm{BF16}}\|}. \tag{14}$$

Following FP4-all-the-way Chmiel et al. (2025), we use $\mathrm{GNR} > \sqrt{3}$ as a theoretically motivated diagnostic reference for assessing whether backward quantization noise is likely to become a limiting

Table 10: Backward GEMM diagnostics on a 500M LLaMA-family model. Kernel speedup is measured for the standalone backward quantization kernel on a $4096 \times 8192$ matrix and normalized to MS-EDEN. It does not measure the speedup of the complete backward GEMM or training step.

| GEMM | Method | Rel-MSE ↓ | GNR ↑ | Margin over $\sqrt{3}$ | Kernel Speedup ↑ |
|---|---|---|---|---|---|
| $\nabla X = EW$ | MS-EDEN | 0.01539 | 8.06 | $4.7\times$ | $1.0\times$ |
| | SR | 0.02181 | 6.77 | $3.9\times$ | $1.5\times$ |
| | Had-SR-16 | 0.02128 | 6.86 | $4.0\times$ | $1.1\times$ |
| $\nabla W = E^\top X$ | MS-EDEN | 0.01043 | 9.79 | $5.7\times$ | $1.0\times$ |
| | SR | 0.01105 | 9.51 | $5.5\times$ | $1.5\times$ |
| | Had-SR-16 | 0.01399 | 8.45 | $4.9\times$ | $1.1\times$ |

factor. This reference follows from a second-order Taylor approximation with zero-mean isotropic stochastic-rounding noise, optimal step-size analysis, and a concentrated-Hessian-spectrum approximation; it should therefore not be interpreted as a universal sufficient condition for convergence. Equivalently, the corresponding relative MSE reference is

$$\mathrm{RelMSE} = \frac{\|G_Q - G_{\mathrm{BF16}}\|^2}{\|G_{\mathrm{BF16}}\|^2} < \frac{1}{3}. \tag{15}$$

Table 10 shows that all evaluated estimators remain well above the GNR diagnostic reference. MS-EDEN further reduces local error, especially for $\nabla X = EW$, but the SR-based estimators already operate in a clearly signal-dominant regime and have faster standalone quantization kernels in our implementation. This suggests that, under the evaluated model scale and training regime, backward quantization error is unlikely to be the dominant source of training-quality degradation. Based on this diagnostic, SCOPE uses a lightweight SR/Had-SR backward path to preserve low-bias estimation, together with deterministic rounding for the relatively static operand to reduce overhead.

**End-to-end validation.** We further validate this interpretation with an end-to-end NanoChat experiment. We replace Had-SR with MS-EDEN, which achieves lower local backward quantization error, and compare full training runs under otherwise identical settings using three matched seeds at the fixed 11B-token endpoint. The results are reported in Table 1 in the main text.

Despite its lower local backward quantization error, MS-EDEN does not yield a lower mean final BPB than Had-SR in these matched runs. Since MS-EDEN is applied throughout training, this comparison gives the lower-error estimator the full optimization horizon to translate its local advantage into a final-task benefit. Together with the GNR diagnostic above, these results support the use of the lightweight Had-SR estimator under the evaluated model scale and training budget: further reducing local backward quantization error does not improve mean final BPB in these experiments.

Additional stochastic-rounding variants explored for the backward path are discussed in Appendix D.2.

### D.2 Stochastic Rounding Variants

We have tried several stochastic rounding variants with the goal of reducing the variance of SR without introducing additional overhead. However, these variants did not yield consistently lower MSE, as shown in Table 11. Among them, we introduce two representative attempts, Antithetic-SR and Systematic-SR.

**Antithetic-SR** pairs adjacent elements and constructs sign-aware negatively correlated rounding errors. For a same-sign pair, the rounding thresholds are $(u, 1-u)$; for an opposite-sign pair, both elements reuse $u$, since the opposite signs reverse one of the signed rounding errors. In both cases, the goal is to partially cancel the pair-wise rounding error in the subsequent dot product.

**Systematic-SR** draws one random offset $u_0 \sim \mathcal{U}(0,1)$ per block and uses

$$u_i = \mathrm{frac}(u_0 + i/16), \qquad i = 0, \ldots, 15. \tag{16}$$

Table 11: Relative MSE of backward GEMMs on a 500M LLaMA-family model. We report the relative MSE for $\nabla X = EW$ and $\nabla W = E^\top X$, together with the relative change with respect to the independent SR baseline.

| GEMM | Method | Rel-MSE ↓ | Δ vs. SR |
|---|---|---|---|
| $\nabla X = EW$ | SR | 0.02181 | – |
| | Antithetic-SR | 0.02182 | ∼0% |
| | Systematic-SR | 0.02181 | ∼0% |
| | Had-SR-16 | 0.02128 | −2.4% |
| | **Had-Systematic-SR-16** | **0.02127** | −2.5% |
| | Had-Antithetic-SR-16 | 0.02136 | −2.1% |
| | MS-EDEN | 0.01539 | −29.4% |
| $\nabla W = E^\top X$ | SR | 0.01105 | – |
| | **Antithetic-SR** | **0.01011** | −8.5% |
| | Systematic-SR | 0.01108 | +0.3% |
| | Had-SR-16 | 0.01399 | +27% |
| | Had-Antithetic-SR-16 | 0.01397 | +26% |
| | Had-Systematic-SR-16 | 0.01403 | +27% |
| | MS-EDEN | 0.01043 | −5.6% |

Thus, the 16 thresholds form a randomly shifted, evenly spaced grid on $[0, 1)$. Each threshold remains marginally uniform, while the construction suppresses clustered rounding decisions and constrains the accumulated block-wise rounding error before GEMM contraction.

For both variants, the stochastic-rounding group size remains 16, matching the NVFP4 block size. The suffix in HAD-SYSTEMATIC-SR-16 or HAD-ANTITHETIC-SR-16 denotes the Hadamard-transform width rather than the SR group size.

Although these variants did not lead to better empirical results, we report them here to document our exploration and potentially avoid duplicated effort in future work.

# E DERIVATION OF THE SCOPE ANALYTICAL SCALE ORACLE

This appendix derives the analytical SCOPE-Oracle used to study the upper bound of adaptive scale selection for NVFP4 forward quantization. The oracle is not used directly as the online training kernel; instead, it provides a block-wise optimal scale under a continuous scale search space and guides the construction of a compact candidate set for the deployed method.

## E.1 NVFP4 BLOCK QUANTIZATION OBJECTIVE

Consider one quantization block

$$x = (x_1, \dots, x_B) \in \mathbb{R}^B, \tag{17}$$

where $B = 16$ for NVFP4. Let

$$M = \max_i |x_i| \tag{18}$$

be the local block maximum. If $M = 0$, the block is identically zero and any scale produces zero reconstruction error; in implementation we simply return $s^\star = 1$ and $\hat{x} = 0$.

For nonzero blocks, SCOPE searches over a scalar scale factor

$$s \in [s_{\min}, s_{\max}]. \tag{19}$$

Given $s$, each element is first normalized as

$$z_i(s) = \frac{6x_i}{Ms}. \tag{20}$$

The constant 6 corresponds to the maximum magnitude in the NVFP4 E2M1 codebook:

$$\mathcal{C} = \{0, \pm 0.5, \pm 1, \pm 1.5, \pm 2, \pm 3, \pm 4, \pm 6\}. \tag{21}$$

Let $Q_{\mathrm{FP4}}(\cdot)$ denote round-to-nearest quantization to $\mathcal{C}$. The dequantized reconstruction is

$$\hat{x}_i(s) = \frac{Ms}{6} Q_{\mathrm{FP4}}\left(\frac{6x_i}{Ms}\right). \tag{22}$$

The block-wise reconstruction error is

$$\mathcal{L}_{\mathrm{q}}(s; x) = \sum_{i=1}^{B} \left(x_i - \frac{Ms}{6} Q_{\mathrm{FP4}}\left(\frac{6x_i}{Ms}\right)\right)^2. \tag{23}$$

The full SCOPE-Oracle solves

$$s^\star = \arg\min_{s \in [s_{\min}, s_{\max}]} \mathcal{L}_{\mathrm{q}}(s; x). \tag{24}$$

### E.2 Rounding Regions and Breakpoints

The positive NVFP4 code values are

$$0,\ 0.5,\ 1,\ 1.5,\ 2,\ 3,\ 4,\ 6. \tag{25}$$

Under round-to-nearest, the positive rounding thresholds are the midpoints between adjacent positive code values:

$$\mathcal{T} = \{0.25,\ 0.75,\ 1.25,\ 1.75,\ 2.5,\ 3.5,\ 5.0\}. \tag{26}$$

Thus, for a nonnegative normalized value $|z|$, the quantized magnitude changes only when $|z|$ crosses one of the thresholds in $\mathcal{T}$.

For a fixed element $x_i$, its normalized magnitude is

$$|z_i(s)| = \left|\frac{6x_i}{Ms}\right| = \frac{6|x_i|}{Ms}. \tag{27}$$

A rounding transition occurs when

$$\frac{6|x_i|}{Ms} = \tau, \qquad \tau \in \mathcal{T}. \tag{28}$$

Solving for $s$ gives the breakpoint induced by element $x_i$ and threshold $\tau$:

$$b_{i,\tau} = \frac{6|x_i|}{M\tau}. \tag{29}$$

Only breakpoints inside the search range can affect the optimum:

$$\mathcal{B} = \{b_{i,\tau} : i = 1, \ldots, B,\ \tau \in \mathcal{T},\ s_{\min} < b_{i,\tau} < s_{\max}\}. \tag{30}$$

We then sort all valid breakpoints and add the two endpoints:

$$\mathcal{P} = \mathrm{sort}\left(\{s_{\min}, s_{\max}\} \cup \mathcal{B}\right). \tag{31}$$

The consecutive pairs in $\mathcal{P}$ partition the search interval into segments.

### E.3 Piecewise-Constant Code Assignment

Consider an interval

$$[s_l, s_r] \tag{32}$$

formed by two consecutive values in $\mathcal{P}$. By construction, there is no breakpoint strictly inside $(s_l, s_r)$. Therefore, for every element $x_i$ and every threshold $\tau \in \mathcal{T}$, the value

$$\frac{6|x_i|}{Ms} \tag{33}$$

does not cross $\tau$ for any $s \in (s_l, s_r)$. Consequently, the quantized code assignment is constant inside the interval:

$$Q_{\mathrm{FP4}}\left(\frac{6x_i}{Ms}\right) = v_i, \qquad \forall s \in (s_l, s_r). \tag{34}$$

In practice, we obtain this fixed assignment by evaluating at the midpoint

$$s_m = \frac{s_l + s_r}{2}, \tag{35}$$

and setting

$$v_i = Q_{\mathrm{FP4}}\left(\frac{6x_i}{Ms_m}\right). \tag{36}$$

**Remark on boundary ties.** At exact breakpoints, an element may lie exactly on a rounding threshold, and the quantized code may depend on the tie-breaking rule. This does not affect the oracle search in practice because we evaluate the final reconstruction error after clipping the candidate scale and re-quantizing at the candidate scale. Thus, even if the midpoint assignment differs from the boundary assignment, the final segment error is computed using the actual quantizer.

### E.4 Closed-Form Optimum Within One Segment

Inside a segment, the code assignment $v_i$ is fixed. Substituting Equation 36 into Equation 23, the segment-wise problem becomes

$$\min_{s\in[s_l,s_r]} \sum_{i=1}^{B} \left( x_i - \frac{Ms}{6} v_i \right)^2. \tag{37}$$

Define

$$u = \frac{Ms}{6}. \tag{38}$$

Then Equation 37 becomes a scalar least-squares problem:

$$\min_{u} \sum_{i=1}^{B} (x_i - uv_i)^2. \tag{39}$$

Expanding the objective gives

$$\sum_i (x_i - uv_i)^2 = \sum_i x_i^2 - 2u \sum_i v_i x_i + u^2 \sum_i v_i^2. \tag{40}$$

Taking the derivative with respect to $u$ yields

$$\frac{\partial}{\partial u} \sum_i (x_i - uv_i)^2 = -2 \sum_i v_i x_i + 2u \sum_i v_i^2. \tag{41}$$

Setting the derivative to zero gives the unconstrained optimum

$$u^\star = \frac{\sum_i v_i x_i}{\sum_i v_i^2}. \tag{42}$$

If $\sum_i v_i^2 = 0$, all elements are quantized to zero inside this segment. In implementation, we optionally use a tiny numerical safeguard without affecting the reported oracle results. In this degenerate case, the reconstruction is zero and the objective is independent of $s$ within the segment; we can choose either endpoint, e.g.,

$$s^\star_{\text{seg}} = s_l. \tag{43}$$

For the non-degenerate case, we map $u^\star$ back to the scale variable:

$$s^\star_{\text{raw}} = \frac{6u^\star}{M}. \tag{44}$$

Because the segment-wise solution must lie inside $[s_l, s_r]$, we project it back:

$$s^\star_{\text{seg}} = \text{clip}\left(s^\star_{\text{raw}}, s_l, s_r\right). \tag{45}$$

### E.5 Segment Error Evaluation

After obtaining $s^\star_{\text{seg}}$, we compute the actual reconstructed block by applying the original FP4 quantizer again:

$$\tilde{v}_i = Q_{\text{FP4}}\left( \frac{6x_i}{M s^\star_{\text{seg}}} \right), \tag{46}$$

and

$$\hat{x}_i^{\text{seg}} = \frac{M s^\star_{\text{seg}}}{6} \tilde{v}_i. \tag{47}$$

The segment reconstruction error is

$$\mathrm{SSE}_{\mathrm{seg}} = \sum_{i=1}^{B} \left(x_i - \hat{x}_i^{\mathrm{seg}}\right)^2. \tag{48}$$

Re-quantizing at $s^\star_{\mathrm{seg}}$ is important for robust handling of clipped solutions and breakpoint ties.

Finally, the full SCOPE-Oracle selects the best segment-wise candidate:

$$s^\star = \arg\min_{s^\star_{\mathrm{seg}}} \mathrm{SSE}_{\mathrm{seg}}. \tag{49}$$

### E.6 Why the Oracle is Exact over the Continuous Scale Range

The objective in Equation 23 is not globally smooth because the quantized code assignment changes discontinuously when a normalized value crosses a rounding threshold. However, all such discontinuities occur at the breakpoints in Equation 29. Between two adjacent breakpoints, the assignment is fixed, and the objective reduces to a convex quadratic function of the scalar variable $u = Ms/6$. Therefore, the minimum over each segment is given by the closed-form least-squares solution in Equation 42, followed by clipping to the segment boundary. Taking the minimum over all segments yields the global optimum over $[s_{\min}, s_{\max}]$.

### E.7 Complexity

For an NVFP4 block of size $B = 16$, there are seven positive rounding thresholds. Thus, the number of raw breakpoints is at most

$$7B = 112. \tag{50}$$

After sorting the breakpoints, the scale interval is partitioned into at most

$$7B + 1 \tag{51}$$

segments. For each segment, SCOPE-Oracle computes one midpoint assignment, one scalar least-squares solution, and one reconstruction error. Thus, the full oracle is tractable as an analysis tool but still too expensive for direct use in every forward pass during pre-training, especially because it requires breakpoint generation and sorting for every block.

### E.8 Connection to SCOPE-Search

The full oracle serves as an upper bound for adaptive scale selection. It reveals how much improvement is possible if each block is allowed to choose its best continuous scale. In the SCOPE-Search method, we do not perform full breakpoint enumeration online. Instead, we use the oracle to identify high-value scale regions and construct a compact candidate set $\mathcal{S}$. At runtime, we perform a lightweight search:

$$s_{\mathrm{SCOPE}}(x) = \arg\min_{s \in \mathcal{S}} \sum_i \left(x_i - \frac{Ms}{6} Q_{\mathrm{FP4}}\left(\frac{6x_i}{Ms}\right)\right)^2. \tag{52}$$

This practical version preserves most of the benefit of the analytical oracle while avoiding its per-block sorting and full segment enumeration overhead.

### E.9 Oracle Upper Bound and Error Propagation

The "upper bound" we refer to in this paper is the upper bound for adaptive scale selection. SCOPE-Oracle strictly optimizes the reconstruction SSE for each NVFP4 block over the continuous scale range. It therefore provides a lower bound on the local reconstruction SSE, or equivalently, an upper bound on the reconstruction improvement recoverable via adaptive scale selection. Importantly, this does not imply an upper bound on downstream task performance, since there is no strict monotonic relationship between local reconstruction SSE and final downstream metrics such as perplexity and accuracy. Therefore, the fact that SCOPE-Oracle may perform worse than SCOPE-Search on certain downstream tasks does not contradict its role as an upper bound for adaptive scale selection.

Table 12: End-to-end reconstruction fidelity on Qwen3-1.7B. Metrics are computed between the final logits of the quantized and BF16 models on WikiText-2 using $256 \times 1024$ tokens. Cosine similarity is computed over concatenated token logits, and MSE is averaged across samples.

| Scheme | Cosine Similarity ↑ | MSE ↓ |
|---|---|---|
| RTN | 0.9522 | 1.6364 |
| Search | 0.9582 | 1.4356 |
| Oracle | **0.9609** | **1.3499** |

By construction, SCOPE-Oracle minimizes the local block-wise SSE. Empirically, this advantage also propagates to lower end-to-end MSE and higher cosine similarity, although these reconstruction metrics are not guaranteed to have a strict or monotonic relationship with downstream task metrics. To examine this distinction, we evaluate the previously discussed Qwen3-1.7B case, where SCOPE-Oracle does not outperform SCOPE-Search in perplexity. As shown in Table 12, both SCOPE-Search and SCOPE-Oracle produce final logits closer to the BF16 reference than RTN, while SCOPE-Oracle achieves the lowest MSE and highest cosine similarity.

## F Scale Dynamics Across Training

To examine whether the candidate-scale distribution remains stable during training, we perform a checkpoint-wise analysis on an archived NanoChat d20 pre-training run (561M parameters, 20 layers, SCOPE forward) at steps 5k, 10k, 20k, and 21.4k, corresponding to 23%, 47%, 93%, and 100% of training, respectively. The validation BPB decreases from 0.8841 to 0.7815 across these checkpoints. At each stage, we recompute the per-block optimal NVFP4 second-level scale for all six projections in layers 0, 10, and 19. Scales are searched over a uniform grid over $[0.5, 3.0]$ with step size 0.1. Activations use the same SHA-256-pinned WikiText-2 probe at every checkpoint. Each stage contains 1.18M weight blocks and 4.72M activation blocks.

Table 14 shows that the aggregate scale distribution is highly stable throughout training. For activations, scale 1.0 remains the most frequent optimum, accounting for approximately 38.1–38.4% of blocks across all checkpoints, while the same four scales $\{1.0, 1.5, 1.4, 0.9\}$ consistently cover approximately 82–83% of blocks. Weights exhibit only a modest shift away from scale 1.0 toward larger scales such as 1.4 and 1.6. Moreover, the reconstruction gap between a compact scale pool and SCOPE-Oracle does not grow during training. For activations, the additional reconstruction SSE decreases from 0.547% to 0.280%, while for weights it decreases from 1.829% to 1.543%. These results indicate that the useful scale regions remain stable throughout training and can therefore be calibrated once and reused.

The stability of the marginal distribution, however, does not imply that individual blocks retain the same optimal scale. To quantify block-wise persistence, we match blocks across checkpoints and report the exact same-scale rate. As a reference, we compute the chance agreement $\sum_i p_i^2$ implied by the marginal scale distribution, and define the lift over chance as the ratio between exact and chance agreement. We also report the mean absolute change in the optimal scale $s^\star$.

As shown in Table 15, over substantial portions of training the per-block scale assignment is only weakly predictable. For activations, the lift over chance is only $1.11$–$1.14\times$, while for weights it is approximately $0.95$–$1.00\times$. Strong persistence appears only between the two nearby late-stage checkpoints, when the learning rate and parameter updates are already small. This indicates that an earlier block-wise scale is not a reliable long-range predictor of the scale preferred later in training.

Overall, this analysis supports SCOPE's construct-then-select design: the marginal scale distribution is stable enough to construct and reuse a fixed candidate pool, whereas the assignment within that pool remains block- and input-dependent and should therefore be selected online. The measurements also suggest that most of the available benefit is concentrated in a small number of scales, while the stronger short-range persistence observed near the end of training may provide an opportunity for future selection-speed optimizations.

Table 13: Task-level accuracy (%) under the NanoChat Speedrun setting. Bold indicates the best result among quantized methods.

| Shot | Dataset | BF16 | 4over6 | NVIDIA | Quartet-II | SCOPE |
|---|---|---|---|---|---|---|
| 0 | hellaswag_zeroshot | 48.47 | 44.85 | 46.40 | 46.54 | **46.86** |
| 0 | copa | 65.33 | 64.67 | 66.00 | **67.33** | 66.00 |
| 0 | openbook_qa | 36.13 | 33.27 | **36.33** | 34.13 | 35.47 |
| 0 | lambada_openai | 40.58 | 36.03 | 37.72 | **40.44** | 39.17 |
| 0 | winograd | 66.54 | **66.79** | 66.54 | 66.67 | **66.79** |
| 0 | winogrande | 55.14 | 54.09 | 54.38 | 54.49 | **54.93** |
| 0 | coqa | 24.02 | 21.29 | **23.32** | **23.32** | 23.04 |
| 0 | **Average** | 48.03 | 45.85 | 47.24 | **47.56** | 47.47 |
| 3 | agi_eval_sat_ar | 25.51 | 26.23 | 24.78 | 24.20 | **27.39** |
| 10 | jeopardy | 12.55 | 9.72 | 8.60 | 7.18 | **10.97** |
| 10 | bigbench_qa_wikidata | 51.20 | 46.08 | 48.00 | **48.88** | 48.79 |
| 10 | arc_easy | 65.64 | 64.62 | 64.77 | 64.51 | **64.81** |
| 10 | arc_challenge | 35.38 | 33.67 | 34.78 | 34.90 | **35.27** |
| 10 | commonsense_qa | 22.36 | 23.53 | **30.00** | 25.33 | 25.09 |
| 10 | piqa | 69.46 | 67.72 | 68.72 | **68.99** | 68.88 |
| 10 | hellaswag | 48.64 | 44.80 | 46.17 | 46.49 | **46.77** |
| 10 | bigbench_dyck_languages | 12.07 | 11.93 | 13.33 | **13.57** | 12.33 |
| 10 | bigbench_cs_algorithms | 37.20 | 37.25 | **41.59** | 35.91 | 38.08 |
| 10 | bigbench_operators | 17.94 | 15.40 | 16.35 | 16.83 | **19.21** |
| 10 | bigbench_repeat_copy_logic | 0.00 | 0.00 | **1.04** | 0.00 | **1.04** |
| 10 | squad | 32.05 | 26.01 | **29.48** | 29.29 | 29.09 |
| 10 | boolq | 56.81 | 49.86 | 50.17 | 51.09 | **55.91** |
| 10 | bigbench_language_identification | 25.41 | **25.55** | 25.19 | 24.95 | 25.19 |
| 10 | **Average** | 34.76 | 32.58 | 34.16 | 33.42 | **34.39** |
| – | **Overall Average** | 38.57 | 36.52 | 37.89 | 37.50 | **38.23** |

# G Generalization Across Architectures, Modules, and Domains

We further evaluate whether SCOPE generalizes beyond the dense language-model settings considered in the main experiments. Specifically, we study (1) MoE models in both post-training quantization and pre-training, (2) attention-specific versus non-attention modules, and (3) vision and vision-language tasks. Across these settings, SCOPE consistently improves over RTN on the corresponding aggregate or reconstruction metrics, suggesting that candidate-based adaptive scale selection is not restricted to a particular model architecture, module type, or task domain.

## G.1 Generalization to MoE Models

We first evaluate SCOPE on Qwen3-30B-A3B, an open-source MoE model, under post-training quantization. As shown in Table 16, SCOPE-Search improves the average downstream accuracy and WikiText perplexity over RTN while also reducing all three expert-routing flip metrics. Although the improvement is not uniform across every individual downstream task, these results indicate that adaptive scale selection remains effective in an MoE architecture and better preserves the router behavior.

We further extend our quantized-training infrastructure from dense Transformers to a 50M-parameter MoE model with Qwen3-MoE-style Transformer blocks. The model is trained for 1.5B tokens using the same weight–activation quantization setting and online candidate-scale selection as in our dense pre-training experiments. As shown in Table 17, SCOPE-Search reduces the BF16 validation-loss gap from 0.039 to 0.033, corresponding to a 15.4% reduction in the quantization-induced loss gap. Although this is a relatively small-scale MoE experiment, it provides complementary evidence that SCOPE can be directly applied to MoE pre-training rather than being limited to dense architectures.

Table 14: Checkpoint-wise statistics of the optimal NVFP4 scales during NanoChat pre-training. Top-4 coverage denotes the fraction of blocks assigned to the four most frequent scales. The SSE gap reports the additional reconstruction error relative to SCOPE-Oracle.

| Operand | Training stage | Most frequent scales (block share) | Coverage | SSE gap ↓ |
|---|---|---|---|---|
| Activation | Early (23%) | 1.0 (38.40%), 1.5 (18.91%), 1.4 (12.60%), 0.9 (12.40%) | 82.3% | 0.547% |
| Activation | Middle (47%) | 1.0 (38.21%), 1.5 (18.90%), 1.4 (13.10%), 0.9 (12.10%) | 82.4% | 0.342% |
| Activation | Late (93%) | 1.0 (38.12%), 1.5 (19.10%), 1.4 (13.30%), 0.9 (12.30%) | 82.8% | 0.231% |
| Activation | Final (100%) | 1.0 (38.27%), 1.5 (19.08%), 1.4 (13.30%), 0.9 (12.20%) | 82.9% | 0.280% |
| Weight | Early (23%) | 1.0 (36.26%), 1.5 (18.04%), 0.9 (14.50%), 1.4 (14.30%) | 83.2% | 1.829% |
| Weight | Middle (47%) | 1.0 (35.75%), 1.5 (18.18%), 0.9 (14.70%), 1.4 (14.40%) | 83.0% | 1.745% |
| Weight | Late (93%) | 1.0 (32.05%), 1.5 (18.09%), 0.9 (15.80%), 1.4 (14.70%) | 80.6% | 1.542% |
| Weight | Final (100%) | 1.0 (31.88%), 1.5 (18.05%), 1.4 (15.80%), 0.9 (14.60%) | 80.4% | 1.543% |

Table 15: Block-wise persistence of the optimal NVFP4 scale across training checkpoints. Chance agreement is the collision probability $\sum_i p_i^2$ implied by the marginal scale distribution. Lift over chance is the ratio between the exact same-scale rate and chance agreement.

| Metric | Activation | | | | Weight | | | |
|---|---|---|---|---|---|---|---|---|
| | $5k \to 10k$ | $10k \to 20k$ | $5k \to 21.4k$ | $20k \to 21.4k$ | $5k \to 10k$ | $10k \to 20k$ | $5k \to 21.4k$ | $20k \to 21.4k$ |
| Training span | 23% | 47% | 77% | 6.5% | 23% | 47% | 77% | 6.5% |
| Exact same-scale rate | 25.03% | 25.23% | 24.52% | 50.03% | 21.49% | 20.60% | 20.31% | 81.65% |
| Chance agreement | 22.14% | 22.07% | 22.14% | 22.15% | 21.43% | 21.19% | 21.43% | 19.30% |
| Lift over chance | 1.13× | 1.14× | 1.11× | 2.26× | 1.00× | 0.97× | 0.95× | 4.23× |
| Mean abs(scale shift) ↓ | 0.2885 | 0.2855 | 0.2910 | 0.1775 | 0.2810 | 0.2838 | 0.2853 | 0.0724 |

### G.2 Generalization Across Transformer Modules

We next study whether the benefit of SCOPE is specific to a particular class of Transformer operators. We separately quantize the attention modules and the remaining modules (primarily MLP layers) of Qwen3-8B. The results are reported in Table 18.

It is worth noting that the two groups contain substantially different numbers of parameters: the non-attention modules contain more parameters than the attention modules, so their absolute degradation should not be directly compared. Within each module group, however, SCOPE-Search consistently improves perplexity, cosine similarity, and MSE over RTN. This suggests that the benefit of adaptive scale selection is not restricted to either attention or MLP-related operators.

Table 16: Post-training quantization results on Qwen3-30B-A3B. Flip rates are measured on WikiText. Bold indicates the best result among quantized methods.

| Metric | BF16 | RTN | SCOPE-search |
|---|---|---|---|
| BoolQ ↑ | 88.750% | **87.951%** | 87.645% |
| ARC-Easy ↑ | 79.340% | 77.820% | **78.914%** |
| ARC-Challenge ↑ | 52.900% | 50.683% | **51.365%** |
| HellaSwag ↑ | 59.550% | **58.634%** | 58.624% |
| Avg. Acc. ↑ | 70.133% | 68.772% | **69.137%** |
| WikiText PPL ↓ | 7.7063 | 8.0217 | **7.9433** |
| Membership Flip ↓ | – | 6.991% | **6.264%** |
| Exact-set Flip ↓ | – | 47.389% | **43.332%** |
| Top-1 Flip ↓ | – | 13.626% | **12.184%** |

Table 17: Quantized pre-training results on a 50M-parameter MoE model trained for 1.5B tokens.

| Scheme | Val Loss ↓ | Loss Increase ↓ |
|---|---|---|
| BF16 | 3.329 | – |
| RTN | 3.368 | 1.17% |
| SCOPE-Search | **3.362** | **0.99%** |

### G.3 Generalization to Vision and Vision-Language Tasks

We further evaluate SCOPE on SigLIP-v2 to study transfer beyond language-only models. We first consider ImageNet-1K by quantizing the encoder linear layers. For the vision-calibrated setting, we use 1,024 images for calibration and the remaining 48,976 images for held-out evaluation. As shown in Table 19, SCOPE-Search improves over RTN with both the language-calibrated and vision-calibrated candidate pools, while vision-specific calibration provides an additional improvement.

To test whether such domain-specific recalibration is necessary, we further evaluate ImageNet-V2, ImageNet-REAL, COCO, and Flickr30K using the same SigLIP-v2 model and the same candidate pool and search configuration without vision-specific recalibration or retuning. SCOPE-Search consistently outperforms RTN across all reported classification and retrieval metrics, providing further evidence that candidate-based scale selection transfers beyond language-only tasks.

Overall, these results complement the dense-language-model experiments in the main paper. SCOPE improves over RTN when applied to MoE models, different Transformer module groups, and vision-language models, and it remains effective in both post-training quantization and pre-training. These experiments suggest that the benefit of candidate-based adaptive scale selection is not tied to a single architecture or task domain.

Table 18: Module-specific PTQ results on Qwen3-8B. Cosine similarity and MSE are computed on WikiText. The number of parameters corresponds to the quantized modules.

| Method | #Params | WikiText PPL ↓ | Cosine Sim. ↑ | MSE ↓ |
|---|---|---|---|---|
| BF16 | – | 8.5932 | 1.00000 | 0 |
| RTN-Attn | 1.51B | 8.5916 | 0.98820 | 0.3979 |
| SCOPE-Search-Attn | 1.51B | **8.5656** | **0.99074** | **0.3234** |
| RTN-Rest | 5.44B | 8.8474 | 0.98382 | 0.5805 |
| SCOPE-Search-Rest | 5.44B | **8.7576** | **0.98749** | **0.4646** |

Table 19: Zero-shot image-classification results (%) with SigLIP-v2. For ImageNet-1K, "lang" and "vision" denote language-calibrated and vision-calibrated candidate pools, respectively. Bold indicates the best quantized result within each dataset.

| Dataset | Scheme | Top-1 ↑ | Real Top-1 ↑ | Strict Top-1 ↑ | Top-5 ↑ |
|---|---|---|---|---|---|
| ImageNet-1K | BF16 | 68.093 | – | – | 86.885 |
| ImageNet-1K | RTN | 65.408 | – | – | 85.260 |
| ImageNet-1K | SCOPE-Search (lang) | 65.428 | – | – | 85.360 |
| ImageNet-1K | SCOPE-Search (vision) | **65.571** | – | – | **85.654** |
| ImageNet-V2 | BF16 | 62.41 | – | – | 81.89 |
| ImageNet-V2 | RTN | 59.45 | – | – | 80.16 |
| ImageNet-V2 | SCOPE-Search | **60.33** | – | – | **80.78** |
| ImageNet-REAL | BF16 | – | 73.83 | 69.69 | 87.87 |
| ImageNet-REAL | RTN | – | 71.43 | 67.09 | 86.42 |
| ImageNet-REAL | SCOPE-Search | – | **72.26** | **67.94** | **86.94** |

Table 20: Zero-shot image-to-text and text-to-image retrieval results (%) with SigLIP-v2. No vision-specific candidate recalibration is used. Bold indicates the best quantized result.

| Dataset | Scheme | I→T | | | T→I | | | mR ↑ |
|---|---|---|---|---|---|---|---|---|
| | | R@1 ↑ | R@5 ↑ | R@10 ↑ | R@1 ↑ | R@5 ↑ | R@10 ↑ | |
| COCO | BF16 | 65.30 | 85.88 | 91.62 | 48.84 | 72.97 | 81.08 | 74.28 |
| COCO | RTN | 62.22 | 83.98 | 89.98 | 46.89 | 71.51 | 79.94 | 72.42 |
| COCO | SCOPE-Search | **63.72** | **84.90** | **90.74** | **47.21** | **71.74** | **80.48** | **73.13** |
| Flickr30K | BF16 | 89.70 | 98.50 | 99.30 | 73.44 | 90.98 | 94.72 | 91.11 |
| Flickr30K | RTN | 86.30 | 97.60 | 98.60 | 70.16 | 88.90 | 93.46 | 89.17 |
| Flickr30K | SCOPE-Search | **87.40** | **97.70** | **99.10** | **71.68** | **89.76** | **94.02** | **89.94** |

# H SCALE-SEARCH OVERHEAD AND SCALING BEHAVIOR

We further analyze the isolated cost of forward scale search and how it scales with hidden size, sequence length, batch size, and model depth. The component-level measurements reported here are performed on GB200. Unless otherwise specified, the Transformer forward-path benchmarks use the default candidate budgets $(K_W, K_A) = (4, 8)$. These measurements isolate forward-path overhead and should be distinguished from the complete-recipe training-time comparison in Section 4.4.

## H.1 ISOLATED FORWARD-PATH MEASUREMENTS

**Quantization-kernel overhead.** Table 21 reports standalone quantization-kernel measurements for the eight-candidate configuration. Across the evaluated tensor shapes, the SCOPE kernel takes 1.17–2.10$\times$ the RTN kernel time. These ratios quantify the cost within the quantization kernel, rather than the overhead relative to a complete Transformer forward pass or training step.

**Transformer forward-path overhead.** Table 22 reports timings for the evaluated Transformer configurations. The additional overhead relative to RTN ranges from 7.8% to 10.9%. Here, the measurement covers the Transformer forward path; it is not the overhead of the complete training recipe. The standalone kernel ratios and the forward-path overhead therefore use different denominators and should not be interpreted interchangeably.

We additionally evaluate large-model-style Transformer-layer configurations. As summarized in Table 23, the additional forward-path overhead is approximately 7% for the LLaMA-3.1-70B-style configuration and below 5% for the 175B-style configuration. Reducing the activation candidate budget from 8 to 4 further lowers the reported overhead to below 3% and below 2%, respectively. These are isolated component-level measurements, not full-model pre-training runs.

## H.2 SCALING ANALYSIS

Let $M = BS$ denote the number of tokens processed by a layer in one forward invocation, where $B$ is the corresponding batch size, $S$ is the sequence length, and $H$ is the hidden size. For fixed architectural ratios, the additional scale-search work per layer is approximately

$$\Delta C_{\text{layer}} = O(K_A MH) + O(K_W H^2), \tag{53}$$

because activation search processes $O(MH)$ elements, whereas weight search processes $O(H^2)$ elements. In comparison, the dominant linear-layer GEMMs require $\Theta(MH^2)$ operations. Consequently, the relative search work scales as

$$\frac{\Delta C_{\text{layer}}}{C_{\text{GEMM}}} = O\left(\frac{K_A}{H}\right) + O\left(\frac{K_W}{BS}\right). \tag{54}$$

This work-based analysis describes the expected amortization mechanisms rather than a hardware-independent latency law. Measured runtime additionally depends on effective throughput, memory traffic, kernel-launch overhead, and implementation details.

**Hidden size.** The activation-search fraction decreases approximately as $1/H$: the number of searched activation elements grows linearly with $H$, whereas linear-layer GEMM work grows quadratically. The weight-search fraction relative to these GEMMs is instead primarily controlled by $M$. Thus, increasing hidden width mainly amortizes the activation-side component.

**Sequence length and batch size.** At fixed hidden size, increasing $M = BS$ amortizes weight search over more tokens. Activation search and linear-layer GEMMs both grow linearly with $M$, so their relative work does not decrease solely because more tokens are processed. However, dense self-attention requires $O(BS^2H)$ work. Increasing sequence length can therefore further reduce the search fraction within the complete forward pass. Here, $B$ refers to the batch processed by the forward invocation, rather than an effective batch size increased only through gradient accumulation.

Table 21: Standalone forward quantization-kernel measurements on GB200. The last column reports the SCOPE kernel latency divided by the RTN kernel latency, using 8 candidates. The results with 4 candidates remain around $1.20\times$ across the evaluated shapes.

| **Tensor shape** | **RTN** ($\mu$s) | **SCOPE / RTN** |
|---|---|---|
| $512 \times 1024$ | 14.26 | $1.19\times$ |
| $1024 \times 5632$ | 15.63 | $1.17\times$ |
| $8192 \times 2048$ | 14.39 | $1.71\times$ |
| $16384 \times 2048$ | 20.50 | $2.10\times$ |
| $16384 \times 5632$ | 57.37 | $1.93\times$ |

Table 22: Transformer forward-path measurements on GB200. $B$, $S$, $H$, and $L$ denote batch size, sequence length, hidden size, and model depth, respectively. Overhead is computed as $T_{\mathrm{SCOPE}}/T_{\mathrm{RTN}} - 1$.

| $B$ | $S$ | $H$ | $L$ | **Params** | **BF16** | **RTN** | **SCOPE** | **Overhead vs. RTN** | **RTN quant. occupancy** |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 512 | 1024 | 5 | 64M | 0.267 | 0.671 | 0.723 | $+7.8\%$ | 64% |
| 1 | 1024 | 2048 | 10 | 514M | 1.305 | 2.167 | 2.403 | $+10.9\%$ | 57% |
| 16 | 512 | 1024 | 5 | 64M | 1.420 | 1.919 | 2.105 | $+9.7\%$ | 41% |
| 16 | 1024 | 2048 | 10 | 514M | 16.236 | 13.845 | 15.094 | $+9.0\%$ | 32% |

Table 23: Reported additional forward-path overhead relative to RTN for large-model-style Transformer-layer configurations. The results are component-level measurements rather than complete training-time comparisons.

| **Configuration** | $K_W = 4$<br>$K_A = 8$ | $K_W = 4$<br>$K_A = 4$ |
|---|---|---|
| LLaMA-3.1-70B-style | $\sim 7\%$ | $< 3\%$ |
| 175B-style ($S = 4$k) | $< 5\%$ | $< 2\%$ |

**Model depth.** For approximately homogeneous Transformer layers, both scale-search work and regular layer computation grow linearly with the number of layers. Model depth therefore largely cancels in the relative overhead. Accordingly, increasing depth alone should leave the overhead fraction approximately unchanged, whereas increasing width or the tokens processed per forward invocation can amortize the corresponding search components.

The analysis charges weight search for each forward invocation and does not assume that weights can be reused across optimizer updates. It also does not require the measured latency overhead to decrease monotonically for every individual tensor shape or batch-size change.

### H.3 Quality–Overhead Trade-off with Training Scale

The training-token budget and the tokens processed in a single forward invocation play different roles. At fixed architecture and per-forward tensor shapes, extending the training-token budget does not itself reduce the relative search work: both regular computation and scale search are repeated over additional training steps.

The corresponding benefit is observed on the quality side. In our controlled forward W/A training ablation, SCOPE's validation-loss advantage over RTN grows with the training-token budget. This observation is consistent with mitigating accumulated forward quantization error, but is not a guarantee of a monotonically increasing advantage at arbitrary training scales.

Together, the observed quality trend and the search-cost analysis provide complementary evidence: wider models and larger per-forward token counts can amortize the relative cost of scale search, while longer training can make its quality benefit more pronounced. The candidate budget provides an

additional quality–overhead control, with $(K_W, K_A) = (4, 8)$ as the default setting and $(K_W, K_A) = (4, 4)$ as the lighter alternative discussed in Appendix B.

Finally, the isolated forward-path overhead should not be added directly to the measured training time of Quartet-II. The complete SCOPE recipe also differs in its backward path. As reported in Section 4.4, the evaluated SCOPE recipe achieves both better validation quality and shorter end-to-end training time than Quartet-II. The scaling analysis explains the cost structure of forward search, while the matched wall-clock experiment evaluates the complete recipe.