RAG-QAF: ROBUST ADAPTATION AND GENERALIZATION FOR QUANTIZATION-AWARE FINE-TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Quantization-aware fine-tuning (QAF) is crucial for adapting pretrained large language models (LLMs) to specific downstream tasks on resource-constrained devices. However, existing QAF methods tend to prioritize in-distribution (ID) performance on target adaptation tasks, while paying limited attention to out-of-distribution (OOD) generalization. Although random parameter perturbations can improve generalization in full-precision fine-tuning, directly introducing such noise into QAF may degrade both ID and OOD performance. In this work, our multi-scale analysis reveals that random noise lacks explicit optimization guidance and overlooks local differences in quantization error. We further observe that the model's global noise sensitivity evolves throughout training, making a fixed perturbation magnitude inadequate. Based on these findings, we propose RAG-QAF, a robust QAF framework for ID adaptation and OOD generalization. Specifically, **local error-guided adversarial optimization** uses quantization errors to guide the search for adversarial perturbations through gradient ascent. **Global adaptive curriculum learning strategy** progressively increases the overall perturbation budget to accommodate evolving noise sensitivity, enabling a transition from stable early optimization to increasingly challenging robust optimization. RAG-QAF can be integrated into existing QAF pipelines as a plug-and-play module. Extensive experiments on ID and OOD benchmarks demonstrate state-of-the-art performance with low additional training overhead.

1 INTRODUCTION

As large language models (LLMs) (Grattafiori et al., 2024; Yang et al., 2025; Glm et al., 2024) are increasingly applied to downstream tasks such as mathematical reasoning (Azerbayev et al., 2024; Yu et al., 2025) and code generation (Sun et al., 2024; Li et al., 2025), resource-efficient deployment and task adaptation have become practical necessities. Although quantization techniques (Shao et al., 2024; Lin et al., 2024) effectively reduce storage and computational costs, the quantized models still require further adaptation to specific tasks. Quantization-aware fine-tuning (QAF) has therefore emerged as an important paradigm for jointly achieving model compression and task adaptation. However, existing QAF methods (Dettmers et al., 2023; Jeon et al., 2025; Li et al., 2024b) primarily focus on in-distribution (ID) performance on target tasks, with limited attention to the out-of-distribution (OOD) generalization of the adapted models.

Prior work on full-precision fine-tuning has explored supervision reweighting and parameter perturbation to mitigate generalization degradation. Token reweighting (Lin et al., 2026; Gong et al., 2026) dynamically adjusts the contribution of individual tokens to the training objective, but may compromise ID performance and incur additional computational overhead. Parameter perturbation methods (Li et al., 2024a; Wu et al., 2022) inject random noise during training to reduce overfitting and improve generalization. However, such noise lacks explicit optimization guidance, and its benefits may depend on the perturbation settings. In QAF, quantization itself already introduces parameter perturbations, making the model's quantization state an additional factor to consider when designing noise. This motivates the following question:

How can the noise perturbation be effectively used in quantization-aware fine-tuning to support both ID adaptation and OOD generalization?

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.

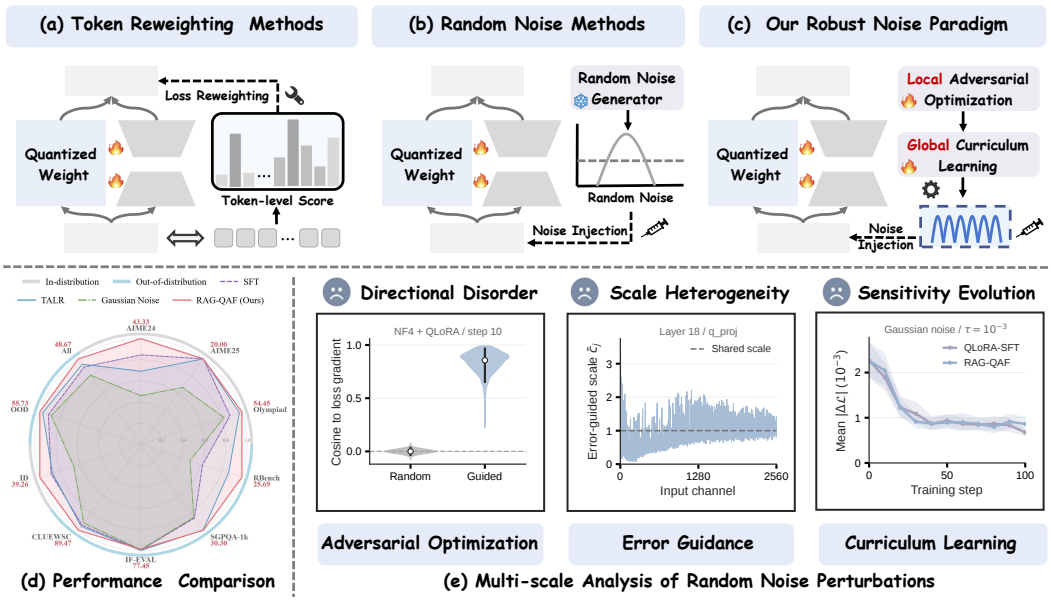


Figure 1: (a–c) Comparison of token reweighting, random noise, and RAG-QAF for quantization-aware fine-tuning. (d) ID and OOD performance comparison on NF4-quantized Qwen3-4B. (e) Multi-scale analysis of gradient alignment, channel-wise error-guided scales, and loss sensitivity to fixed-magnitude Gaussian noise across training.

Firstly, we attempt to improve generalization by injecting random noise into model parameters during quantization-aware fine-tuning. However, as shown in Fig. 1(d), this strategy reduces both average ID and OOD performance relative to SFT on NF4-quantized Qwen3-4B. To understand this phenomenon, we further analyze the effects of noise perturbations on quantized models from a multi-scale perspective, as illustrated in Fig. 1(e). **At the local level**, random noise lacks explicit optimization guidance. Meanwhile, as shown in the middle plot of Fig. 1(e), we observe substantial heterogeneity in channel-wise error-guided scales, indicating that a shared scale cannot capture these local differences. Consequently, random perturbations that ignore channel differences may unnecessarily disrupt relatively stable regions, compromising general knowledge retained from pre-training and thereby weakening OOD generalization. **At the global level**, as shown in the right plot of Fig. 1(e), we observe that fixed-magnitude Gaussian noise induces larger absolute loss changes early in training, with smaller changes at later stages. This evolving sensitivity motivates a progressively increasing perturbation budget to balance stable early adaptation with more challenging robust optimization later in training.

Based on these analyses, we propose RAG-QAF, a robust quantization-aware fine-tuning framework for ID adaptation and OOD generalization that can be integrated into existing QAF pipelines as a plug-and-play module. To account for the characteristics of **local** perturbations, we develop local error-guided adversarial optimization, which uses quantization errors to guide the search for adversarial perturbations through gradient ascent. These adversarial perturbations enhance model robustness, supporting both ID adaptation and OOD generalization. Motivated by the observed changes in **global** noise sensitivity, we introduce a global adaptive curriculum learning strategy. This curriculum gradually increases the overall perturbation budget throughout training, enabling a transition from stable early optimization to increasingly challenging robust optimization. Our contributions can be summarized as follows:

- We systematically analyze the characteristics of noise optimization in quantization-aware fine-tuning from a novel multi-scale perspective. Locally, random noise lacks explicit optimization guidance, and quantization errors vary substantially in magnitude across channels. Globally, the model's sensitivity to noise evolves throughout training.
- We propose RAG-QAF, a robust quantization-aware fine-tuning framework for ID adaptation and OOD generalization that combines local error-guided adversarial optimization

with global adaptive curriculum learning to account for local noise characteristics and evolving global noise sensitivity, respectively.

- Extensive experiments demonstrate that RAG-QAF consistently improves the trade-off between in-distribution adaptation and out-of-distribution generalization, while introducing low additional training overhead.

2 Related Work

2.1 Robustness of Quantized Large Language Models

The robustness of quantized large language models (LLMs) refers to their ability to maintain stable performance under distribution shifts after quantization. Existing LLM quantization methods (Liu et al., 2025; Shao et al., 2024) typically rely on calibration data to estimate quantization parameters or weight importance, and thus primarily optimize performance under the calibration distribution. GPTQ (Frantar et al., 2022) reconstructs quantized weights layer by layer using approximate second-order information, SmoothQuant (Xiao et al., 2023) mitigates activation outliers through activation statistics, and AWQ (Lin et al., 2024) preserves a small subset of salient weights based on activation statistics. Recent studies (Marchisio et al., 2024; Mekala et al., 2025) have identified challenges in generalizing quantized models across different languages and evaluation settings. LRQuant (Zhao et al., 2024) reports limited generalization to unseen datasets when models are quantized using a fixed calibration set. Similarly, NSNQuant (Son et al., 2026) shows that KV-cache quantization methods relying on calibration data to learn codebooks can exhibit limited transferability when moving from in-distribution (ID) to out-of-distribution (OOD) data. These findings suggest that preserving performance on the standard distribution does not necessarily guarantee OOD robustness after quantization. Meanwhile, QLoRA (Dettmers et al., 2023) and QA-LoRA (Xu et al., 2024) incorporate quantization into downstream task adaptation to improve the performance of quantized models on specific downstream tasks. However, these methods primarily focus on performance within the target task distribution, with limited consideration of robustness under distribution shifts. Therefore, our work investigates how to further improve both ID and OOD performance, building on the benefits of quantized fine-tuning to enhance model robustness across different distributions.

2.2 Quantization-Aware Fine-Tuning

Quantization-aware fine-tuning aims to adapt low-bit models to downstream tasks while compensating for quantization-induced discrepancies through parameter optimization. QLoRA (Dettmers et al., 2023) enables low-rank adapter training on a frozen 4-bit quantized backbone, substantially reducing the cost of fine-tuning large language models. LoftQ (Li et al., 2024b) jointly initializes quantized weights and low-rank adapters to reduce the discrepancy introduced by quantization and improve downstream adaptation. These methods combine low-bit model representations with parameter-efficient fine-tuning, enabling effective task adaptation with reduced training and deployment costs. Recent studies (Liao et al., 2024; Wei et al., 2025) further couple quantization with task-specific optimization. LQ-LoRA (Guo et al., 2024) represents model weights using quantized and low-rank components and updates only the low-rank components during adaptation, keeping the quantized components fixed. L4Q (Jeon et al., 2025) directly incorporates low-rank updates into quantization-aware training, allowing the adapted weights to be merged and deployed as a low-bit model. However, existing methods mainly optimize target-distribution performance, leaving room to further improve OOD generalization alongside ID adaptation. Our method uses the model's actual quantization error as a key signal during quantization-aware fine-tuning and optimizes its perturbation effect through adversarial training, further improving both ID performance and OOD generalization.

3 Methodology

In this section, we provide an overview of RAG-QAF, as illustrated in Fig. 2. Local Error Guided Adversarial Optimization (Sec. 3.1) uses quantization errors to guide adversarial search in the normalized channel space. Global Adaptive Curriculum Learning Strategy (Sec. 3.2) progressively increases the noise budget over training.

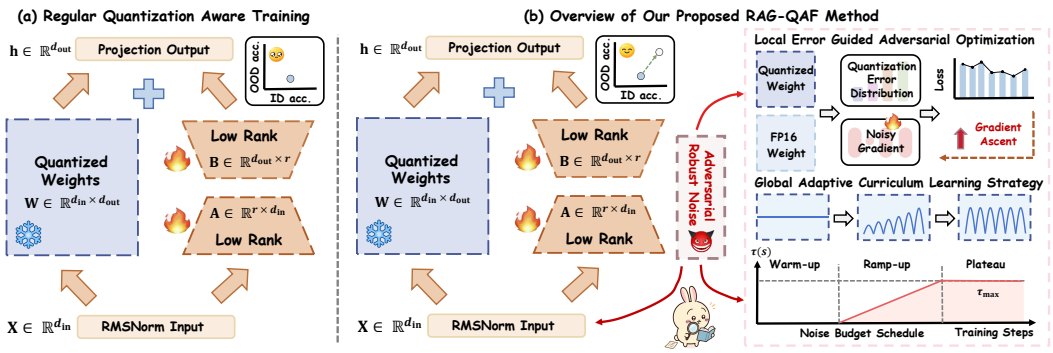


Figure 2: (a) Standard quantization-aware fine-tuning with frozen quantized weights and trainable low-rank adapters. (b) Overview of RAG-QAF. Quantization errors guide channel-wise perturbation search in Local Error Guided Adversarial Optimization, while Global Adaptive Curriculum Learning Strategy progressively increases the noise budget. Snowflakes and flames denote frozen and trainable components, respectively.

3.1 Local Error Guided Adversarial Optimization

Local Adversarial Optimization. To enhance robustness during fine-tuning, we optimize adversarial noise in the normalized channel space through gradient ascent. Let $\widehat{\Theta}$ and ϕ denote the frozen quantized backbone parameters and the trainable LoRA parameters, respectively. For an input-response pair $(x, y) \sim \mathcal{D}_{\text{train}}$, the supervised fine-tuning loss is

$$\mathcal{L}_{\text{SFT}}(\phi, \mathcal{V}; x, y) = -\sum_{t \in \mathcal{I}_y} \log p_{\widehat{\Theta},\phi}\big(y_t \mid x, y_{<t}; \mathcal{V}\big). \tag{1}$$

Here, $\mathcal{I}_y$ indexes the supervised response tokens, excluding prompt and padding positions. Let $\mathcal{V}$ denote the collection of layer-specific channel-wise noise vectors $v \in \mathbb{R}^d$, where d is the number of input channels in the corresponding layer. Each vector is shared across the batch and sequence dimensions.

We add adversarial noise v to the frozen RMSNorm scaling parameter $\gamma \in \mathbb{R}^d$. For a single token with pre-normalization activation $a \in \mathbb{R}^d$, the normalized representation u and the perturbed input h to the linear layer are given by

$$u = \frac{a}{\sqrt{\frac{1}{d}\sum_{j=1}^{d} a_j^2 + \epsilon_{\text{norm}}}}, \qquad h = (\gamma + v) \odot u. \tag{2}$$

Here, $\odot$ denotes element-wise multiplication, and $\epsilon_{\text{norm}} > 0$ is a numerical stability term. The noise vector v modulates the channel magnitudes of the normalized representation before it enters the current linear layer, allowing gradient ascent to directly search for challenging directions in the channel space. For the corresponding linear layer, let $\widehat{W}, \Delta W(\phi) \in \mathbb{R}^{m \times d}$ denote the frozen quantized weight and the LoRA weight update, respectively, where m is the number of output channels. Both act on the same perturbed input h, yielding the output $z \in \mathbb{R}^m$ as

$$z = \widehat{W}h + \Delta W(\phi)h. \tag{3}$$

Thus, the gradient with respect to v depends on the quantized weight and LoRA update. Each noise vector has only d entries, resulting in low additional training overhead. For any $q \in \mathbb{R}^n$, we define $\text{RMS}(q) = \sqrt{\frac{1}{n}\sum_{j=1}^{n} q_j^2}$. We independently constrain each noise vector as follows.

$$\mathcal{B}(\tau) = \{\mathcal{V} : \forall v \in \mathcal{V},\ \text{RMS}(v) \le \tau\}. \tag{4}$$

At optimizer step s, we use a noise budget of $\tau(s)$ to bound the RMS magnitude of each noise vector. Under this constraint, the training objective is formulated as

$$\min_{\phi} \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{train}}} \left[\max_{\mathcal{V}\in\mathcal{B}(\tau(s))} \mathcal{L}_{\text{SFT}}(\phi, \mathcal{V}; x, y) \right]. \tag{5}$$

We update the LoRA parameters through gradient descent to learn in-distribution (ID) task knowledge, while gradient ascent searches within the noise budget for channel-wise perturbations that increase the loss. Minimizing the task loss under these perturbations encourages the model to adapt to ID data while reducing its sensitivity to local channel variations, thereby promoting out-of-distribution (OOD) generalization.

Local Quantization Error Guidance. Standard adversarial optimization primarily relies on task gradients, while the local impact of quantization varies across input channels. To use local quantization errors as an optimization signal, we quantify their relative impact on each channel from the weight differences before and after quantization and use this information to reweight the adversarial gradient. For a linear layer, let $W \in \mathbb{R}^{m\times d}$ denote its full-precision reference weight before quantization. The quantization error is defined as

$$E = W - \widehat{W}. \tag{6}$$

For input channel j, we first normalize its quantization error magnitude $\|E_{:,j}\|_2$ by the corresponding quantized weight magnitude $\|\widehat{W}_{:,j}\|_2$. We then incorporate the RMSNorm scaling magnitude $|\gamma_j|$ to compute c_j, which characterizes the relative impact of quantization on this channel. We collect these values into $c = (c_1, \ldots, c_d)^\top$ and apply RMS normalization as follows.

$$c_j = |\gamma_j| \frac{\|E_{:,j}\|_2}{\|\widehat{W}_{:,j}\|_2 + \epsilon_w}, \qquad \bar{c} = \frac{c}{\text{RMS}(c) + \epsilon_c}. \tag{7}$$

Here, $\epsilon_w, \epsilon_c > 0$ ensure numerical stability in the denominators. We compute $\bar{c}$ before training and keep it fixed throughout optimization.

During training, we evaluate the SFT loss at step s using the current LoRA parameters $\phi^{(s)}$ and noise state $\mathcal{V}^{(s)}$. We then compute its gradient with respect to the noise vector v as

$$g^{(s)} = \nabla_v \mathcal{L}_{\text{SFT}}\left(\phi^{(s)}, \mathcal{V}^{(s)}; x, y\right). \tag{8}$$

Then, we reweight the noise gradient $g^{(s)}$ element-wise using the precomputed $\bar{c}$ to account for the different local effects of quantization across channels. We apply RMS normalization to the weighted gradient to obtain the adversarial ascent direction $d^{(s)} \in \mathbb{R}^d$ for the subsequent noise update.

$$d^{(s)} = \frac{\bar{c} \odot g^{(s)}}{\text{RMS}(\bar{c} \odot g^{(s)}) + \epsilon_g}. \tag{9}$$

Here, $\epsilon_g > 0$ stabilizes the denominator in the normalization above. To optimize the inner maximization objective, we take a gradient ascent step along the adversarial direction $d^{(s)}$ to search for perturbations that increase the current loss. We set the step size to $\kappa\tau_{\max}$, using a fixed coefficient $\kappa = 0.1$ and a default maximum noise budget $\tau_{\max} = 10^{-3}$. We then project the updated noise, yielding

$$v^{(s+1)} = \Pi_{\tau(s)}\left(v^{(s)} + \kappa\tau_{\max} d^{(s)}\right), \tag{10}$$

where $\Pi_{\tau(s)}$ denotes the Euclidean projection onto the set $\{v \in \mathbb{R}^d : \text{RMS}(v) \le \tau(s)\}$. We apply this update independently to each noise vector. We adjust the budget $\tau(s)$ over training using the global adaptive curriculum learning strategy described in Sec. 3.2.

3.2 Global Adaptive Curriculum Learning Strategy

In the early stages of training, the model is still adapting to the ID distribution, and strong adversarial noise may disrupt this process. We therefore adopt a progressive curriculum that gradually relaxes the noise magnitude constraint to balance early ID adaptation with subsequent robust optimization for OOD generalization.

Specifically, at each optimization step s, we apply the projection in Sec. 3.1 to constrain each noise vector to satisfy $\mathrm{RMS}(v) \le \tau(s)$, where $\tau(s)$ controls the maximum allowed noise magnitude. We initialize all noise vectors to zero and first perform noise-free ID adaptation for S_0 steps. We then linearly increase the noise magnitude limit to $\tau_{\max}$ over S_{ramp} steps and keep this limit fixed for the rest of training. The corresponding noise budget is defined as

$$\tau(s) = \begin{cases} 0, & s < S_0, \\ \tau_{\max} \dfrac{s - S_0 + 1}{S_{\mathrm{ramp}}}, & S_0 \le s < S_0 + S_{\mathrm{ramp}}, \\ \tau_{\max}, & s \ge S_0 + S_{\mathrm{ramp}}. \end{cases} \tag{11}$$

Under this curriculum, we search for challenging perturbation directions within progressively expanding local neighborhoods. Once the budget reaches $\tau_{\max}$, we continue updating the adversarial noise under the fixed magnitude constraint for the remainder of training. When noise is enabled, each microbatch uses one backward pass of the SFT loss to compute both LoRA and noise gradients. After gradient accumulation, we update LoRA parameters once by gradient descent and noise vectors once by quantization-error-guided gradient ascent followed by projection. The updated noise vectors are retained for the next optimizer step, requiring no additional forward or backward pass. It is notable that we use curriculum scheduling and noise optimization only during training. At inference, we retain only the quantized backbone and trained LoRA parameters, introducing no additional inference overhead.

4 Experiments

4.1 Experimental Setup

Models and Baselines. We apply RAG-QAF to pretrained LLMs, including the GLM4 (Glm et al., 2024) and Qwen3 (Yang et al., 2025) families. These models are representative open-source LLMs from different model families, enabling evaluation across different architectures. We compare our method with standard Supervised Fine-Tuning (SFT) and two types of baselines that aim to balance in-distribution (ID) performance and out-of-distribution (OOD) generalization. The first type consists of token reweighting methods, including TALR (Lin et al., 2026), FLOW (Sanyal et al., 2025), VCORE (Gong et al., 2026), and DFT (Wu et al., 2026), which adjust the contribution of individual training tokens during fine-tuning to improve model generalization. For the second type, we construct a Gaussian Noise baseline by introducing random Gaussian perturbations during fine-tuning.

Datasets and Evaluation. For the primary experiments, we fine-tune models on a publicly released mathematical reasoning dataset (Gong et al., 2026). We evaluate in-distribution (ID) performance on AIME 2024 (Balunovic et al., 2026), AIME 2025 (Balunovic et al., 2026), and OlympiadBench (He et al., 2024). For out-of-distribution (OOD) evaluation, we use IFEval (Zhou et al., 2023), CLUEWSC (Xu et al., 2020), RBench (Guo et al., 2025), and SGPQA-1k (Du et al., 2026). To assess the applicability of RAG-QAF across fine-tuning domains, we further evaluate it on code generation and biomedical tasks; the corresponding training datasets and evaluation benchmarks are detailed in Sec. 4.4.

Experimental Details. We implement RAG-QAF with Hugging Face (Wolf et al., 2020) and PyTorch (Paszke et al., 2019). Experiments were conducted on AMD Instinct MI308X 192GB GPUs. We adopt NF4 quantization (Dettmers et al., 2023) and a QLoRA-based fine-tuning framework as our default experimental setup. We further evaluate RAG-QAF under MXFP4 (Rouhani et al., 2023) and MXINT4 (Chen et al., 2025) quantization in Tables 2 and 3 and with LoftQ (Li et al., 2024b) and EoRA (Liu et al., 2024) in Fig. 7 to assess its applicability.

Table 1: Main results on in-distribution (ID) and out-of-distribution (OOD) benchmarks. Boldface and underlining indicate the best and second-best scores within each model's NF4 block; for ties, only the bottommost entry is marked.

Format	Method	In-distribution ↑			Out-of-distribution ↑				Average ↑		
		AIME24	AIME25	Olympiad	RBench	SGPQA-1k	IF-EVAL	CLUEWSC	ID	OOD	All
Qwen3-4B											
BF16	–	16.67	13.33	48.37	18.37	31.30	78.56	87.17	26.12	53.85	41.97
NF4	–	20.00	13.33	41.84	12.07	23.50	78.37	82.89	25.06	49.21	38.86
NF4	SFT (Ouyang et al., 2022)	36.67	20.00	48.07	15.72	26.00	78.00	84.21	34.91	50.98	44.10
NF4	TALR (Lin et al., 2026)	30.00	20.00	53.12	22.30	30.20	**78.37**	85.53	34.37	54.10	45.65
NF4	DFT (Wu et al., 2026)	26.67	23.33	54.30	24.50	**31.40**	77.45	86.84	34.77	55.05	46.36
NF4	FLOW (Sanyal et al., 2025)	33.33	20.00	54.15	19.01	28.00	76.16	**90.13**	35.83	53.33	45.83
NF4	VCORE (Gong et al., 2026)	26.67	**26.67**	49.11	16.64	26.80	78.19	83.88	34.15	51.38	43.99
NF4	Gaussian Noise	20.00	13.33	44.81	12.52	25.70	77.63	81.91	26.05	49.44	39.42
NF4	**RAG-QAF (Ours)**	**43.33**	20.00	**54.45**	**25.69**	30.30	77.45	89.47	**39.26**	**55.73**	**48.67**
Qwen3-8B											
BF16	–	23.33	13.33	49.26	19.84	35.30	81.89	86.84	28.64	55.97	44.26
NF4	–	20.00	23.33	49.55	16.91	32.30	82.07	81.25	30.96	53.13	43.63
NF4	SFT (Ouyang et al., 2022)	33.33	23.33	50.89	21.02	33.80	81.89	81.91	35.85	54.66	46.60
NF4	TALR (Lin et al., 2026)	26.67	23.33	51.93	23.31	**36.50**	**83.73**	81.91	33.98	56.36	46.77
NF4	DFT (Wu et al., 2026)	30.00	20.00	51.93	22.03	35.40	82.44	81.91	33.98	55.45	46.24
NF4	FLOW (Sanyal et al., 2025)	33.33	20.00	53.41	22.21	33.30	81.70	82.24	35.58	54.86	46.60
NF4	VCORE (Gong et al., 2026)	36.67	26.67	55.64	28.24	35.70	82.62	83.55	39.66	57.53	49.87
NF4	Gaussian Noise	36.67	30.00	56.38	27.61	34.70	82.07	83.55	41.02	56.98	50.14
NF4	**RAG-QAF (Ours)**	**40.00**	**36.67**	**56.38**	**30.53**	36.20	82.62	**84.21**	**44.35**	**58.39**	**52.37**
GLM-4-9B											
BF16	–	10.00	6.67	39.91	38.39	30.40	80.59	92.43	18.86	60.45	42.63
NF4	–	6.67	6.67	38.87	37.66	28.00	79.30	92.76	17.40	59.43	41.42
NF4	SFT (Ouyang et al., 2022)	6.67	10.00	37.98	38.12	28.30	79.30	93.09	18.22	59.70	41.92
NF4	TALR (Lin et al., 2026)	6.67	6.67	38.13	42.23	**29.10**	78.74	92.76	17.16	60.71	42.04
NF4	DFT (Wu et al., 2026)	13.33	3.33	37.39	**42.23**	29.00	79.85	93.09	18.02	**61.04**	42.60
NF4	FLOW (Sanyal et al., 2025)	6.67	0.00	38.72	36.75	27.00	79.48	92.43	15.13	58.92	40.15
NF4	VCORE (Gong et al., 2026)	3.33	6.67	35.61	38.85	28.20	79.85	93.09	15.20	60.00	40.80
NF4	Gaussian Noise	6.67	6.67	37.54	37.02	28.30	79.48	93.09	16.96	59.47	41.25
NF4	**RAG-QAF (Ours)**	**20.00**	**10.00**	**39.91**	37.39	28.80	**79.85**	**93.09**	**23.30**	59.78	**44.15**

4.2 Main Results

In this section, we first evaluate RAG-QAF on the Qwen3 and GLM model families. As shown in Table 1, RAG-QAF achieves the highest overall average performance across the in-distribution (ID) and out-of-distribution (OOD) benchmarks among the compared methods. For example, RAG-QAF achieves relative improvements of 8.12% and 1.49% in average ID and OOD performance on Qwen3-8B over the respective best baselines, Gaussian Noise and VCORE. These improvements in overall performance across model families suggest that RAG-QAF helps balance task adaptation and OOD generalization during quantization-aware fine-tuning.

4.3 Ablation Study

Component Ablation. We evaluate the two modules through cumulative ablations of adversarial optimization (AO) and quantization error guidance (QEG) within the local module, followed by the global curriculum learning (CL) strategy. Fig. 3 reports the results on Qwen3-4B and Qwen3-8B. On Qwen3-8B, adding channel-wise adversarial optimization to standard supervised fine-tuning (SFT) improves average ID and OOD performance from 35.85 and 54.66 to 39.12 and 57.60, respectively, indicating benefits for both task adaptation and OOD generalization. Quantization error guidance further improves these scores to 41.46 and 58.31, supporting the use of actual quantization errors to guide adversarial updates. Incorporating the global curriculum strategy further improves both ID and OOD performance, validating the effectiveness of progressively increasing the noise budget. Qwen3-4B exhibits the same trend, with overall average performance increasing from 44.10 under SFT to 48.67 with the full method.

Robustness to the Noise Hyperparameter $\tau_{\max}$. We vary $\tau_{\max}$ while keeping $\kappa = 0.1$ fixed. As shown in Fig. 4, varying the maximum noise budget hyperparameter $\tau_{\max}$ changes the average score

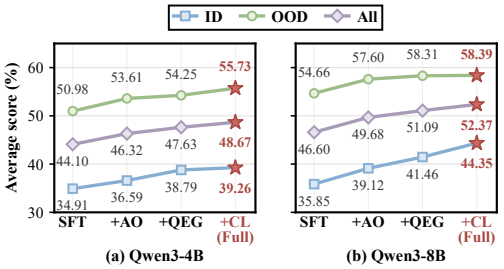


Figure 3: Cumulative component ablation on Qwen3-4B and Qwen3-8B.

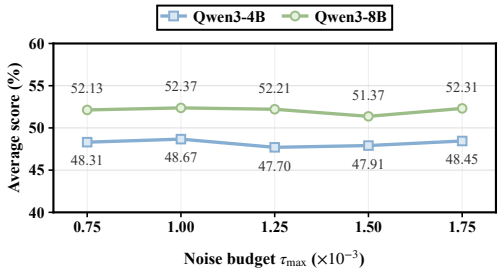


Figure 4: Sensitivity to the maximum noise budget on Qwen3-4B and Qwen3-8B.

Table 2: MXFP4 results on Qwen3-8B.

Format	Method	ID	OOD	All
BF16	–	28.64	55.97	44.26
MXFP4	–	30.00	55.11	44.35
	SFT (Ouyang et al., 2022)	31.34	55.28	45.02
	TALR (Lin et al., 2026)	32.59	**57.14**	46.62
	DFT (Wu et al., 2026)	38.40	56.69	48.85
	FLOW (Sanyal et al., 2025)	28.22	55.80	43.98
	VCORE (Gong et al., 2026)	30.25	53.99	43.81
	Gaussian Noise	37.38	55.97	48.00
	RAG-QAF (Ours)	**42.62**	57.06	**50.87**

Table 3: MXINT4 results on Qwen3-8B.

Format	Method	ID	OOD	All
BF16	–	28.64	55.97	44.26
MXINT4	–	28.27	52.61	42.18
	SFT (Ouyang et al., 2022)	29.21	54.41	43.61
	TALR (Lin et al., 2026)	29.51	54.90	44.02
	DFT (Wu et al., 2026)	32.81	56.02	46.07
	FLOW (Sanyal et al., 2025)	29.70	55.03	44.17
	VCORE (Gong et al., 2026)	29.60	53.22	43.10
	Gaussian Noise	29.04	54.15	43.39
	RAG-QAF (Ours)	**33.51**	**56.08**	**46.41**

across seven benchmarks by at most 0.97 and 1.00 percentage points on Qwen3-4B and Qwen3-8B, respectively. Both models outperform their respective best baselines at every tested budget, indicating that RAG-QAF maintains stable gains without relying on a precise value of this hyperparameter.

Effectiveness Across Quantization Formats. We further evaluate RAG-QAF on Qwen3-8B under MXFP4 and MXINT4 quantization. As shown in Tables 2 and 3, RAG-QAF consistently improves upon standard supervised fine-tuning (SFT) under the same quantization format. The relative gains in average ID and OOD performance are 35.99% and 3.22% under MXFP4 and 14.72% and 3.07% under MXINT4, respectively. RAG-QAF also achieves the highest overall average performance among the compared methods under both formats. These results indicate that the benefits of RAG-QAF extend beyond the NF4 format used in the main experiments.

4.4 Efficiency and Generalization Analysis

Training Time and Peak Memory. Compared with the evaluated token reweighting methods for improving generalization, RAG-QAF incurs less additional training time, as its LoRA and adversarial noise gradients are computed in the same forward and backward pass. To demonstrate this advantage, we compare average training step time and peak GPU memory on NF4-quantized Qwen3-4B trained in parallel on four AMD Instinct MI308X OAM GPUs, each with 192GB of available memory. As shown in Fig. 5, RAG-QAF adds only 2.15% training time over SFT, compared with 25.78% for TALR and 124.82% for VCORE, while maintaining comparable peak memory. Combined with the performance results in Table 1, these results show that RAG-QAF achieves better ID adaptation and OOD generalization with limited additional resource overhead.

Scalability to Larger Models. We further evaluate RAG-QAF on Qwen3-32B under NF4 quantization to examine whether its gains persist at a larger model scale. As shown in Fig. 6, RAG-QAF achieves average ID, OOD, and overall scores of 46.30, 67.20, and 58.24, respectively, outperforming all compared baselines on these metrics. Relative to SFT under the same quantization setting, the average ID and OOD scores improve by 35.50% and 3.53%, respectively, while overall average performance increases from 51.74 to 58.24. These results are consistent with the trends observed on smaller models, indicating that RAG-QAF continues to benefit both task adaptation and OOD generalization at the 32B scale.

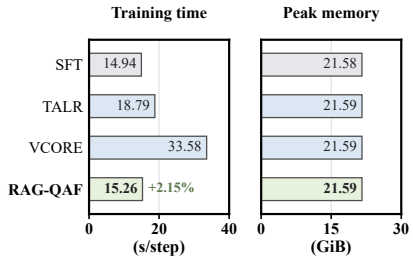


Figure 5: Training time and peak memory on Qwen3-4B.

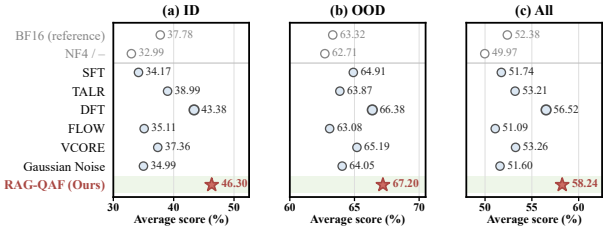


Figure 6: ID adaptation and OOD generalization on Qwen3-32B under NF4 quantization.

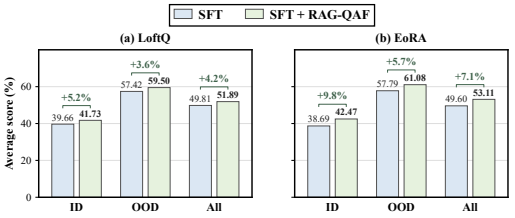


Figure 7: Effectiveness of RAG-QAF with LoftQ and EoRA on Qwen3-8B.

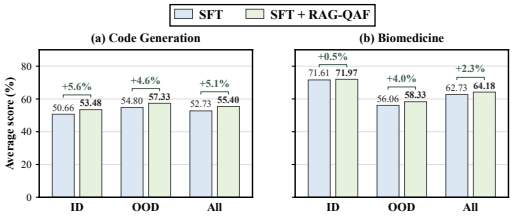


Figure 8: Effectiveness of RAG-QAF in code generation and biomedical fine-tuning.

Compatibility with Different Quantization-Aware Fine-Tuning Methods. The preceding experiments primarily adopt the widely used QLoRA framework for quantization-aware fine-tuning. To examine whether the benefits of RAG-QAF depend on this framework, we integrate RAG-QAF into quantized fine-tuning pipelines based on LoftQ (Li et al., 2024b) and EoRA (Liu et al., 2024) and compare each with its corresponding configuration without RAG-QAF. As shown in Fig. 7, RAG-QAF consistently improves performance across these fine-tuning methods, supporting its compatibility and general applicability.

Generality across Fine-Tuning Domains. In this section, we extend our experiments to the code and biomedical domains. We use the C++ subset of OpenCodeReasoning-2 (Ahmad et al., 2025) for code fine-tuning and MedThoughts-8K for biomedical fine-tuning. The in-distribution (ID) benchmarks comprise OJBench (Wang et al., 2025), LiveCodeBench v6 (Jain et al., 2025), MBPP (Austin et al., 2021), and HumanEval+ (Liu et al., 2023) for the code domain, and MedMCQA (Pal et al., 2022), MedQA (Jin et al., 2021), and PubMedQA (Jin et al., 2019) for the biomedical domain. We evaluate out-of-distribution (OOD) performance using IFEval, CLUEWSC, RBench, and SGPQA-1k in both domains. As shown in Fig. 8, under quantization-aware fine-tuning of Qwen3-8B, RAG-QAF achieves relative improvements of 5.6% and 4.6% in average ID and OOD performance, respectively, over standard supervised fine-tuning (SFT) in the code domain, with corresponding improvements of 0.5% and 4.0% in the biomedical domain. These consistent gains across both domains indicate that the benefits of RAG-QAF extend beyond a single fine-tuning domain.

5 Conclusion

In this work, we investigate how noise perturbations can improve both in-distribution (ID) adaptation and out-of-distribution (OOD) generalization in quantization-aware fine-tuning (QAF). Our multi-scale analysis reveals that random perturbations are insufficient to accommodate local differences in quantization error and evolving global noise sensitivity. Based on these observations, we propose RAG-QAF, a robust QAF framework combining local error-guided adversarial optimization with global adaptive curriculum learning. The framework uses quantization errors to guide the search for local adversarial perturbations, while progressively increasing the overall perturbation budget to accommodate changes in noise sensitivity throughout training. This design supports robust adaptation under increasingly challenging perturbations while maintaining optimization stability. Extensive experiments on ID and OOD benchmarks demonstrate state-of-the-art performance with low additional computational overhead. Code will be released upon acceptance.

AI USE STATEMENT

In this work, we used generative AI tools primarily for language editing, including improving clarity, grammar, and presentation. All AI-assisted content was carefully reviewed, verified, and revised by the authors. The research ideas, methodology, experimental design, results, and conclusions were developed and validated by the authors. We take full responsibility for the final content of this work.

REFERENCES

Wasi Uddin Ahmad, Somshubra Majumdar, Aleksander Ficek, Sean Narenthiran, Mehrzad Samadi, Jocelyn Huang, Siddhartha Jain, Vahid Noroozi, and Boris Ginsburg. Opencodereasoning-ii: A simple test time scaling approach via self-critique. *arXiv preprint arXiv:2507.09075*, 2025.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Qiaochu Jiang, Jia Deng, Stella R Biderman, and Sean Welleck. Llemma: An open language model for mathematics. In *International Conference on Learning Representations*, volume 2024, pp. 40622–40649, 2024.

Mislav Balunovic, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions. *Advances in Neural Information Processing Systems*, 38, 2026.

Mengzhao Chen, Meng Wu, Hui Jin, Zhihang Yuan, Jing Liu, Chaoyi Zhang, Yunshui Li, Jie Huang, Jin Ma, Zeyue Xue, et al. Int vs fp: A comprehensive study of fine-grained low-bit quantization formats. *arXiv preprint arXiv:2510.25602*, 2025.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

Xeron Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, Chujie Zheng, et al. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. *Advances in Neural Information Processing Systems*, 38, 2026.

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.

Team Glm, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*, 2024.

Xuan Gong, Senmiao Wang, Hanbo Huang, Ruoyu Sun, and Shiyu Liang. Vcore: Variance-controlled optimization-based reweighting for chain-of-thought supervision. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28161–28180, 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Han Guo, Philip Greengard, Eric Xing, and Yoon Kim. Lq-lora: Low-rank plus quantized matrix decomposition for efficient language model finetuning. In *International Conference on Learning Representations*, volume 2024, pp. 50167–50184, 2024.

Meng-Hao Guo, Jiajun Xu, Yi Zhang, Jiaxi Song, Haoyang Peng, Yi-Xuan Deng, Xinzhi Dong, Kiyohiro Nakayama, Zhengyang Geng, Chen Wang, et al. R-bench: Graduate-level multi-disciplinary benchmarks for llm & mllm complex reasoning evaluation. *arXiv preprint arXiv:2505.02018*, 2025.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024.

Naman Jain, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, volume 2025, pp. 58791–58831, 2025.

Hyesung Jeon, Yulhwa Kim, and Jae-Joon Kim. L4q: parameter efficient quantization-aware fine-tuning on large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2002–2024, 2025.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. Pubmedqa: A dataset for biomedical research question answering. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 2567–2577, 2019.

Haochen Li, Wanjin Feng, Xin Zhou, and Zhiqi Shen. Gift: Gibbs fine-tuning for code generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12271–12284, 2025.

Tao Li, Qinghua Tao, Weihao Yan, Zehao Lei, Yingwen Wu, Kun Fang, Mingzhen He, and Xiaolin Huang. Revisiting random weight perturbation for efficiently improving generalization. *arXiv preprint arXiv:2404.00357*, 2024a.

Yixiao Li, Yifan Yu, Chen Liang, Nikos Karampatziakis, Pengcheng He, Weizhu Chen, and Tuo Zhao. Loftq: Lora-fine-tuning-aware quantization for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 13409–13424, 2024b.

Baohao Liao, Christian Herold, Shahram Khadivi, and Christof Monz. Apiq: Finetuning of 2-bit quantized large language model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 20996–21020, 2024.

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.

Jiacheng Lin, Zhongruo Wang, Kun Qian, Tian Wang, Arvind Srinivasan, Hansi Zeng, Ruochen Jiao, Xie Zhou, Jiri Gesi, Dakuo Wang, et al. Sft doesn't always hurt general capabilities: Revisiting domain-specific fine-tuning in llms. In *International Conference on Learning Representations*, volume 2026, pp. 46954–46992, 2026.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in neural information processing systems*, 36:21558–21572, 2023.

Shih-Yang Liu, Maksim Khadkevich, Nai Chit Fung, Charbel Sakr, Chao-Han Huck Yang, Chien-Yi Wang, Saurav Muralidharan, Hongxu Yin, Kwang-Ting Cheng, Jan Kautz, et al. Eora: Fine-tuning-free compensation for compressed llm with eigenspace low-rank approximation. *arXiv preprint arXiv:2410.21271*, 2024.

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.

Kelly Marchisio, Saurabh Dash, Hongyu Chen, Dennis Aumiller, Ahmet Üstün, Sara Hooker, and Sebastian Ruder. How does quantization affect multilingual llms? In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 15928–15947, 2024.

Anmol Mekala, Anirudh Atmakuru, Yixiao Song, Marzena Karpinska, and Mohit Iyyer. Does quantization affect models' performance on long-context tasks? In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 9433–9481, 2025.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pp. 248–260. PMLR, 2022.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Bita Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, et al. Microscaling data formats for deep learning. *arXiv preprint arXiv:2310.10537*, 2023.

Sunny Sanyal, Hayden Prairie, Rudrajit Das, Ali Kavis, and Sujay Sanghavi. Upweighting easy samples in fine-tuning mitigates forgetting. *arXiv preprint arXiv:2502.02797*, 2025.

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.

Donghyun Son, Euntae Choi, and Sungjoo Yoo. Nsnquant: A double normalization approach for calibration-free low-bit vector quantization of kv cache. *Advances in Neural Information Processing Systems*, 38:43124–43159, 2026.

Tao Sun, Linzheng Chai, Jian Yang, Yuwei Yin, Hongcheng Guo, Jiaheng Liu, Bing Wang, Liqun Yang, and Zhoujun Li. Unicoder: Scaling code large language model via universal code. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1812–1824, 2024.

Zhexu Wang, Yiping Liu, Yejie Wang, Wenyang He, Bofei Gao, Muxi Diao, Yanxu Chen, Kelin Fu, Flood Sung, Zhilin Yang, et al. Ojbench: A competition level code benchmark for large language models. *arXiv preprint arXiv:2506.16395*, 2025.

Quan Wei, Chung-Yiu Yau, Hoi-To Wai, Yang Katie Zhao, Dongyeop Kang, Youngsuk Park, and Mingyi Hong. Roste: An efficient quantization-aware supervised fine-tuning approach for large language models. *arXiv preprint arXiv:2502.09003*, 2025.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, 2020.

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. Noisytune: A little noise can help you finetune pretrained language models better. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 680–685, 2022.

Yongliang Wu, Yizhou Zhou, Ziheng Zhou, Yingzhe Peng, Xinyu Ye, Xinting Hu, Wenbo Zhu, Lu Qi, Ming-Hsuan Yang, et al. On the generalization of sft: A reinforcement learning perspective with reward rectification. In *International Conference on Learning Representations*, volume 2026, pp. 27550–27571, 2026.

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.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, et al. Clue: A chinese language understanding evaluation benchmark. In *Proceedings of the 28th international conference on computational linguistics*, pp. 4762–4772, 2020.

Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhengsu Chen, Xiaopeng Zhang, and Qi Tian. Qa-lora: Quantization-aware low-rank adaptation of large language models. In *International Conference on Learning Representations*, volume 2024, pp. 52401–52418, 2024.

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.

Yiyao Yu, Yuxiang Zhang, Dongdong Zhang, Xiao Liang, Hengyuan Zhang, Xingxing Zhang, Mahmoud Khademi, Hany Hassan Awadalla, Junjie Wang, Yujiu Yang, et al. Chain-of-reasoning: Towards unified mathematical reasoning in large language models via a multi-paradigm perspective. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 24914–24937, 2025.

Jiaqi Zhao, Miao Zhang, Chao Zeng, Ming Wang, Xuebo Liu, and Liqiang Nie. Lrquant: Learnable and robust post-training quantization for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2240–2255, 2024.

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

Table 4: Per-benchmark MXFP4 results on Qwen3-8B.

Format	Method	In-distribution ↑			Out-of-distribution ↑				Average ↑		
		AIME24	AIME25	Olympiad	RBench	SGPQA-1k	IF-EVAL	CLUEWSC	ID	OOD	All
BF16	–	23.33	13.33	49.26	19.84	35.30	81.89	86.84	28.64	55.97	44.26
MXFP4	–	26.67	13.33	50.00	18.92	32.10	80.59	88.82	30.00	55.11	44.35
	SFT (Ouyang et al., 2022)	26.67	13.33	54.01	23.13	32.60	78.56	86.84	31.34	55.28	45.02
	TALR (Lin et al., 2026)	20.00	23.33	54.45	24.86	**35.80**	80.41	87.50	32.59	**57.14**	46.62
	DFT (Wu et al., 2026)	33.33	26.67	55.19	25.50	34.90	79.85	86.51	38.40	56.69	48.85
	FLOW (Sanyal et al., 2025)	20.00	13.33	51.34	20.11	33.10	**81.15**	**88.82**	28.22	55.80	43.98
	VCORE (Gong et al., 2026)	26.67	13.33	50.74	18.10	32.60	78.74	86.51	30.25	53.99	43.81
	Gaussian Noise	33.33	23.33	55.49	24.41	33.10	79.85	86.51	37.38	55.97	48.00
	RAG-QAF (Ours)	**33.33**	**36.67**	**57.86**	**27.33**	35.30	79.11	86.51	**42.62**	57.06	**50.87**

Table 5: Per-benchmark MXINT4 results on Qwen3-8B.

Format	Method	In-distribution ↑			Out-of-distribution ↑				Average ↑		
		AIME24	AIME25	Olympiad	RBench	SGPQA-1k	IF-EVAL	CLUEWSC	ID	OOD	All
BF16	–	23.33	13.33	49.26	19.84	35.30	81.89	86.84	28.64	55.97	44.26
MXINT4	–	23.33	16.67	44.81	13.35	29.60	80.96	86.51	28.27	52.61	42.18
	SFT (Ouyang et al., 2022)	23.33	16.67	47.63	16.36	32.90	**81.52**	86.84	29.21	54.41	43.61
	TALR (Lin et al., 2026)	26.67	13.33	48.52	18.28	33.20	80.96	87.17	29.51	54.90	44.02
	DFT (Wu et al., 2026)	23.33	**23.33**	51.78	21.48	**35.30**	80.78	86.51	32.81	56.02	46.07
	FLOW (Sanyal et al., 2025)	20.00	20.00	49.11	18.01	34.30	80.96	86.84	29.70	55.03	44.17
	VCORE (Gong et al., 2026)	20.00	20.00	48.81	14.53	31.30	80.22	86.84	29.60	53.22	43.10
	Gaussian Noise	20.00	16.67	50.45	16.54	31.60	80.96	87.50	29.04	54.15	43.39
	RAG-QAF (Ours)	**26.67**	20.00	**53.86**	**24.86**	33.40	78.56	**87.50**	**33.51**	**56.08**	**46.41**

A Appendix Overview

Appendix B provides the per-benchmark results underlying the aggregate comparisons reported in the main paper. Appendix C summarizes the training configuration, evaluation settings, and Gaussian Noise baseline configuration.

B Detailed Experimental Results

The following tables provide the per-benchmark results underlying the quantization-format, model-scale, fine-tuning-method, and cross-domain evaluations in the main paper. ID and OOD are the unweighted means over their respective benchmark groups, and All is the unweighted mean over all displayed benchmarks. Boldface and underlining mark the best and second-best distinct scores within each low-bit comparison block, excluding BF16 references. As in Table 1, only the bottom-most entry is marked in a tie.

B.1 Results across Quantization Formats

Tables 4 and 5 expand Tables 2 and 3, respectively, with the seven individual benchmark scores on Qwen3-8B. RAG-QAF achieves the highest overall average under both formats. Under MXFP4, TALR has a slightly higher OOD average, while RAG-QAF obtains the highest ID average. Under MXINT4, RAG-QAF obtains the highest ID and OOD averages, although individual benchmarks favor different methods.

B.2 Results on Qwen3-32B

Table 6 provides the per-benchmark results corresponding to Fig. 6. RAG-QAF obtains the highest ID, OOD, and overall averages within the NF4 block. Its strongest results occur on the three mathematical reasoning benchmarks, RBench, and SGPQA-1k. SFT retains the highest IFEval and CLUEWSC scores in this block, indicating that the aggregate gains do not imply improvements on every individual benchmark.

Table 6: Per-benchmark results on Qwen3-32B under NF4 quantization.

Format	Method	In-distribution ↑			Out-of-distribution ↑				Average ↑		
		AIME24	AIME25	Olympiad	RBench	SGPQA-1k	IF-EVAL	CLUEWSC	ID	OOD	All
BF16	–	40.00	23.33	50.00	35.19	45.10	84.84	88.16	37.78	63.32	52.38
NF4	–	26.67	23.33	48.96	34.64	45.00	83.36	87.83	32.99	62.71	49.97
	SFT (Ouyang et al., 2022)	23.33	26.67	52.52	39.95	46.50	**85.03**	**88.16**	34.17	64.91	51.74
	TALR (Lin et al., 2026)	36.67	30.00	50.30	38.30	46.00	83.36	87.83	38.99	63.87	53.21
	DFT (Wu et al., 2026)	40.00	33.33	56.82	45.80	47.40	84.47	87.83	43.38	66.38	56.52
	FLOW (Sanyal et al., 2025)	30.00	26.67	48.66	35.37	45.20	83.92	87.83	35.11	63.08	51.09
	VCORE (Gong et al., 2026)	36.67	23.33	52.08	41.59	47.40	83.92	87.83	37.36	65.19	53.26
	Gaussian Noise	30.00	23.33	51.63	39.21	45.80	83.36	87.83	34.99	64.05	51.60
	RAG-QAF (Ours)	**43.33**	**36.67**	**58.90**	**49.36**	**49.00**	82.62	87.83	**46.30**	**67.20**	**58.24**

Table 7: Per-benchmark results with LoftQ and EoRA on Qwen3-8B.

Method	In-distribution ↑			Out-of-distribution ↑				Average ↑		
	AIME24	AIME25	Olympiad	RBench	SGPQA-1k	IF-EVAL	CLUEWSC	ID	OOD	All
LoftQ (Li et al., 2024b)										
SFT (Ouyang et al., 2022)	33.33	30.00	**55.64**	27.61	35.00	83.18	83.88	39.66	57.42	49.81
RAG-QAF (Ours)	**33.33**	**36.67**	55.19	**34.46**	**36.30**	**83.36**	**83.88**	**41.73**	**59.50**	**51.89**
EoRA (Liu et al., 2024)										
SFT (Ouyang et al., 2022)	33.33	26.67	56.08	30.53	35.00	80.41	**85.20**	38.69	57.79	49.60
RAG-QAF (Ours)	**40.00**	**30.00**	**57.42**	**38.39**	**39.10**	**82.62**	84.21	**42.47**	**61.08**	**53.11**

Table 8: Code generation results on Qwen3-8B under NF4 quantization.

Format	Method	In-distribution ↑				Out-of-distribution ↑				Average ↑		
		LCB v6	OJBench	HumanEval+	MBPP	RBench	SGPQA-1k	IF-EVAL	CLUEWSC	ID	OOD	All
NF4	SFT (Ouyang et al., 2022)	45.21	5.17	78.05	74.20	21.21	33.50	82.26	82.24	50.66	54.80	52.73
	RAG-QAF (Ours)	**49.95**	**6.47**	**81.10**	**76.40**	**26.97**	**37.50**	**82.26**	**82.57**	**53.48**	**57.33**	**55.40**

Table 9: Biomedical results on Qwen3-8B under NF4 quantization.

Format	Method	In-distribution ↑			Out-of-distribution ↑				Average ↑		
		MedMCQA	MedQA	PubMedQA	RBench	SGPQA-1k	IF-EVAL	CLUEWSC	ID	OOD	All
NF4	SFT (Ouyang et al., 2022)	62.78	76.75	**75.30**	23.03	34.90	82.44	83.88	71.61	56.06	62.73
	RAG-QAF (Ours)	**63.76**	**77.06**	75.10	**28.98**	**37.10**	**83.36**	**83.88**	**71.97**	**58.33**	**64.18**

B.3 Results with LoftQ and EoRA

Table 7 expands the paired comparisons in Fig. 7 on NF4-quantized Qwen3-8B. Each block compares standard SFT and RAG-QAF within the same fine-tuning pipeline. RAG-QAF improves the ID, OOD, and overall averages with both LoftQ and EoRA. The per-benchmark scores also show the small OlympiadBench decrease with LoftQ and the CLUEWSC decrease with EoRA.

B.4 Results across Fine-Tuning Domains

Tables 8 and 9 provide the per-benchmark results for the SFT and SFT + RAG-QAF comparisons in Fig. 8, using Qwen3-8B under NF4 quantization. The code ID average covers four benchmarks, and the biomedical ID average covers three; both domains use the same four OOD benchmarks as the mathematical reasoning setting. LCB v6 denotes LiveCodeBench v6.

Adding RAG-QAF to SFT improves both ID and OOD averages in each domain. In code generation, the ID and OOD averages increase from 50.66 and 54.80 to 53.48 and 57.33, respectively. In biomedicine, they increase from 71.61 and 56.06 to 71.97 and 58.33, respectively, matching the aggregate comparisons in Fig. 8.

Table 10: Training configuration for mathematical fine-tuning of Qwen3-8B.

Configuration	**Setting**
Base model	Qwen3-8B
Training data	VCORE-data; 3,200 mathematical reasoning samples
Maximum sequence length	16,384 tokens
LoRA configuration	`all-linear`; rank $= 8$, alpha $= 16$, dropout $= 0.1$
Optimizer	AdamW; $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, weight decay $= 0$
Learning rate and schedule	2×10^{-5}; cosine schedule with 10 warmup steps
Gradient clipping threshold	1
Training steps	100
Batch configuration	4 GPUs; microbatch size 1 per GPU; 8 gradient accumulation steps; effective batch size 32
Random seed	42
Noise-free stage	$S_0 = 10$ steps
Budget ramp length	$S_{\mathrm{ramp}} = 60$ steps
Maximum noise RMS budget	$\tau_{\max} = 10^{-3}$
Noise budget schedule	No noise at steps 1–10; linear increase to 10^{-3} at steps 11–70; fixed budget at steps 71–100
Adversarial ascent step size	$\alpha_{\mathrm{adv}}(s) = \kappa\tau(s)$, with $\kappa = 0.1$

Table 11: Evaluation and Gaussian Noise baseline settings.

Configuration	**Setting**
Evaluation model	Training perturbations disabled; LoRA merged
Inference and decoding	vLLM; fixed prompts; greedy decoding with temperature $= 0$ and seed $= 42$
Mathematics, R-Bench-T, SuperGPQA	Thinking enabled; maximum output length of 8,192 tokens
IFEval, CLUEWSC	Thinking disabled; maximum output length of 2,048 tokens
Gaussian Noise	Zero-mean Gaussian noise sampled independently for each projection and resampled at every microbatch. We evaluate $\sigma \in \{10^{-4}, 5 \times 10^{-4}, 10^{-3}\}$, fixed within each run, and report the best performance among the three settings.

C Implementation Details

Table 10 reports the RAG-QAF configuration for mathematical fine-tuning of Qwen3-8B. Table 11 summarizes the evaluation settings and Gaussian Noise baseline configuration. Training steps in this section refer to optimizer steps and are numbered from one.