QUANTLRM: QUANTIZATION OF LARGE REASONING MODELS VIA FINE-TUNING SIGNALS

Anonymous authors
Paper under double-blind review

ABSTRACT

Weight-only quantization is important for compressing Large Reasoning Models (LRMs). However, existing post-training quantization (PTQ) methods are sub-optimal for LRMs that are typically fine-tuned from base models, because they leave their fine-tuning traces wasted during PTQ. Inspired by the spirit of classical magnitude pruning, we study whether the magnitude of weight updates during reasoning-incentivized fine-tuning can provide valuable signals for quantizing LRMs. We hypothesize that the smallest and largest weight updates are more important than those of intermediate magnitude, a phenomenon we term "protecting both ends". Upon hypothesis validation, we introduce QuantLRM, the first method for weight quantization of LRMs that explicitly leverages fine-tuning signals. We fit simple restricted quadratic functions on weight updates to protect both ends. By multiplying the average quadratic values with the count of zero weight updates of channels, we compute channel importance that is more effective than using activation or second-order information, which guides quantization by protecting important weights. To demonstrate the effectiveness and applicability of QuantLRM, we evaluate sub-4-bit quantization performance on various fine-tuning strategies (SFT, DPO, and RLVR) over different model families (Llama, Qwen, and Olmo) on a range of reasoning benchmarks (AIME-120, FOLIO, temporal sequences, and GPQA-Diamond). QuantLRM delivers consistent gains in LRMs quantization, improving average accuracy by up to 6.55% over the best baselines.[1]

1 INTRODUCTION

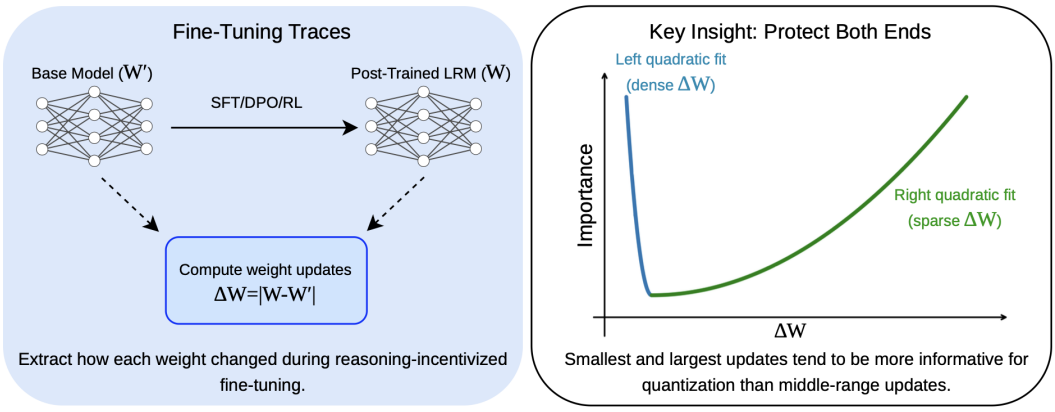


Figure 1: Core idea of QuantLRM. We first gather fine-tuning traces ΔW by taking the absolute value between post-trained LRM and its base model. To protect weight updates on both ends, we fit restricted quadratic functions on ΔW to handle uneven distributions of both sides.

[1]Our code is provided in supplementary material and will be released upon acceptance.

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.

Weight-only quantization is critical for improving the efficiency of Large Reasoning Models (LRMs), enabling the deployment of compact and efficient models for downstream applications. Modern LRMs are typically derived from base Large Language Models (LLMs) through extensive fine-tuning to incentivize reasoning capabilities (Zhang et al., 2026; Liu et al., 2025; Olmo et al., 2025; Yang et al., 2025; Guo et al., 2025). To compress such models without retraining, existing post-training quantization (PTQ) methods, such as AWQ (Lin et al., 2024) and GPTQ (Frantar et al., 2022), rely primarily on information from the final model state, including weight values, activations, or local curvature approximations, to localize important weights and guide quantization.

However, this reliance on the final model state makes existing PTQ methods suboptimal for LRMs. In particular, they disregard the rich signals during fine-tuning, which encode valuable information about weight importance. Importantly, such fine-tuning signals are readily available in practice without requiring modifications to standard training pipelines (e.g., via intermediate checkpoints or differences from the base model). This limitation is especially pronounced in low-bit quantization settings, where accurate weight importance estimation is crucial (Goel et al., 2025).

To address this research gap, we first conduct preliminary experiments in Section 3.1 to understand whether the weight updates during reasoning-incentivized fine-tuning can provide effective signals for quantizing LRMs. We empirically observe that the magnitude of weight updates alone is insufficient, indicating that large updates do not fully capture parameter importance. This phenomenon differs from the common practice of treating large-magnitude values as important, such as large second-order sensitivity (Frantar et al., 2022) and activations (Lin et al., 2024). We then hypothesize that *both the smallest and largest updates carry the most informative signals, as they reflect complementary roles in the model*. Parameters with minimal updates are the ones that are well-optimized before fine-tuning, while parameters with large updates are those most responsible for adapting the model to reasoning tasks. Our preliminary results in Table 1 validate our hypothesis.

Following our hypothesis validation, we propose QuantLRM, the first method for weight quantization of LRMs that explicitly leverages fine-tuning signals, as illustrated in Figure 1. Specifically, we first fit two piecewise quadratic functions to map weight change magnitudes to their importance scores, assigning higher importance to both ends. We then define channel importance by multiplying the average quadratic mapping score by the count of zero-valued weight updates in this channel. The importance reflects the sensitivity of model performance to perturbations of the corresponding weights, and is used to prioritize their preservation during quantization by assigning larger penalties to their quantization errors. This formulation protects weight updates on both ends while also emphasizing parameters with negligible updates. We inject the resulting computed channel importance scores into the quantization loss function as scaling factors and search for the optimal scaling magnitude to minimize the loss. With minimal adaptation on different models, QuantLRM supports mainstream LRMs and achieves state-of-the-art reasoning performance after quantization.

To evaluate the effectiveness of our approach, we use QuantLRM to quantize three model families (Llama (Grattafiori et al., 2024), Olmo (Olmo et al., 2025), Qwen (Yang et al., 2025)) with various fine-tuning strategies (supervised fine-tuning (SFT), direct preference optimization (DPO), and reinforcement learning with verifiable rewards (RLVR)), evaluated over four reasoning benchmarks (AIME-120 (Mathematical Association of America), FOLIO (Han et al., 2024), temporal sequences of BIG-Bench Hard (Suzgun et al., 2022), GPQA-Diamond (Rein et al., 2024)). Our results show that QuantLRM consistently outperforms the strongest PTQ baselines (AWQ, GPTQ, ANY3 (Elhoushi & Johnson, 2025), and SliderQuant (Wang et al., 2026)) on W3A16, achieving an average improvement of 6.55% on an RL fine-tuned model and at least 1.65% on several SFT LRMs. This demonstrates the value of leveraging fine-tuning signals to guide LRM quantization. Compatible with vLLM (Kwon et al., 2023) and AWQ kernel, QuantLRM offers a comparable speedup to state-of-the-art PTQ.

Our **main contributions** are threefold: (1) We introduce a new perspective on LRMs quantization by leveraging fine-tuning signals, hypothesizing and empirically verifying that the smallest and largest weight updates during fine-tuning are more important for quantization than those of intermediate magnitude; (2) We propose QuantLRM, the first PTQ method that explicitly incorporates fine-tuning signals, using a simple yet effective piecewise quadratic mapping to preserve informative signals at both ends; (3) We conduct extensive experiments across multiple LRMs families, fine-tuning paradigms, and reasoning benchmarks, demonstrating that QuantLRM consistently achieves state-of-the-art performance, with substantial improvements over strong PTQ baselines.

2 Related Work

Post-Training Quantization (PTQ). Without retraining, PTQ often requires a small calibration dataset to reach quantization decisions via forward passes only. As recent representative approaches, AWQ uses activation distribution (Lin et al., 2024); GPTQ uses approximate second-order information (Frantar et al., 2022); GPTAQ designs optimized calibration between the outputs of quantized layers and full-precision model (Li et al., 2025); ANY4 (or ANY3 for 3-bit) computes a non-uniform 4-bit numeric representation (Elhoushi & Johnson, 2025); and SliderQuant (Wang et al., 2026) uses an adaptive sliding quantization concept facilitated by few learnable parameters. A drawback is that all these methods are originally designed for general-purpose LLMs, not LRMs. Recent benchmarking analyses (Zhang et al., 2026; Liu et al., 2025) establish that although they achieve almost lossless performance on 4-bit LRMs, their scores on 3-bit LRMs are suboptimal. Selecting all of them as baselines, we demonstrate that fine-tuning signals yield significantly better performance on 3-bit quantization (an important result for deploying 3-bit LRMs). Mixed-precision methods (Ping et al., 2024; Pham et al., 2025) are not studied due to their hardware inefficiency.

Fine-Tuning Traces in LLMs. Except a recent method for 8-bit quantization (Yu et al., 2026) that leverages fine-tuning traces, traces left by fine-tuning are mainly studied for interpretation and safety purposes. For example, understanding weight updates in natural language makes interpreting training processes easier (Goel et al., 2025). Fine-tuning a BERT-size model (Devlin et al., 2019) predominantly alters a low-dimensional subspace of the model's parameters (Zhang et al., 2023), which implies weight importance. However, this method does not provide fine-grained importance scores used for weight-only compression. Moreover, it is computationally more expensive than QuantLRM when applied to LRMs due to its adoption of singular value decomposition. Truthfulness and refusal behaviors are encoded in specific weight updates (Du et al., 2025), which reinforces our motivation of using fine-tuning signals. Regarding safety, works have been done to detect whether a fine-tuned model is pursuing unintended objectives by training with hidden behaviors (Marks et al., 2025) and to determine if a model has been adversarially fine-tuned (Egler et al., 2025).

3 Methodology

In this section, we describe the proposed QuantLRM method that explicitly leverages fine-tuning signals. We first present and empirically validate our hypothesis of protecting weight updates on both ends when computing channel importance (Section 3.1). We then formalize our findings into function mapping from weight updates to importance scores to guide quantization (Section 3.2).

3.1 Overlooked Value of Fine-Tuning Traces

Hypothesis. We hypothesize that the weight change during reasoning-incentivized fine-tuning indicates weight importance for quantization: parameters with the smallest and largest updates carry most informative signals and thus should have large importance score to be protected during quantization. Formally, given a fine-tuned LRM with individual weight $\mathbf{w}_{cm\ell}$ in channel c (*e.g.*, input or output channel) of module m (*e.g.*, self-attention query projection) at layer ℓ, and its value in the base model as $\mathbf{w}'_{cm\ell}$, the weight update $\Delta\mathbf{w}_{cm\ell}$ strongly correlates to its weight importance $\mathbf{i}_{cm\ell}$ for quantization:

$$\Delta\mathbf{w}_{cm\ell} = |\mathbf{w}_{cm\ell} - \mathbf{w}'_{cm\ell}|, \quad \mathbf{i}_{cm\ell} = f(\Delta\mathbf{w}_{cm\ell}) \tag{1}$$

where $f(\cdot)$ maps the weight change to the importance score.

Hypothesis Validation. To empirically validate our hypothesis, we perform proof-of-concept experiments in Table 1. We evaluate ideas via mixed-precision quantization: protecting a fraction of weights based on channel importance in 16-bit while performing round-to-nearest 3-bit quantization on the rest of weights in linear layers. On channel c of module m at layer ℓ, the channel importance $\mathbf{I}_{cm\ell}$ is defined by average importance of individual weights in the channel:

$$\mathbf{I}_{cm\ell} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{i}^{(n)}_{cm\ell} \tag{2}$$

where N is the number of weights. As a baseline following AWQ (Lin et al., 2024), we use activation as the protection signal: $\mathbf{i}_{cm\ell} = (\mathbf{a}_{cm\ell})^2$, where $\mathbf{a}_{cm\ell}$ is the average activation of $\mathbf{w}_{cm\ell}$ estimated on a general-purpose or reasoning-centric calibration dataset (first two rows in Table 1).

Table 1: Validation of "protecting both ends" hypothesis. We perform mixed-precision quantization based on various protection signals (3-bit with top 5% of weights in 16-bit). Protecting both the smallest and largest weight updates ("SFT both ends") yields the best overall performance.

Model	Protection Signal	Accuracy				
		AIME-120	FOLIO	Temporal	GPQA-Diamond	Avg
`R1-Distill-Qwen-32B`	Activation w/ general-purpose calibration	44.2	73.9	99.6	**56.1**	68.45
`R1-Distill-Qwen-32B`	Activation w/ reasoning calibration	40.8	72.4	**100.0**	54.0	66.80
`R1-Distill-Qwen-32B`	SFT high	34.2	70.4	99.2	50.5	63.58
`R1-Distill-Qwen-32B`	SFT mid	38.3	72.4	99.6	50.0	65.08
`R1-Distill-Qwen-32B`	SFT low	38.3	75.4	100.0	54.0	66.93
`R1-Distill-Qwen-32B`	SFT both ends	**49.2**	**77.8**	99.6	54.5	**70.28**

We first check whether higher $\Delta\mathbf{w}_{cm\ell}$ means higher $\mathbf{i}_{cm\ell}$: $\mathbf{i}_{cm\ell} = \Delta\mathbf{w}_{cm\ell}$. This tests whether the absolute values of weight updates can guide quantization decisions, similar to classical magnitude pruning (Han et al., 2015). The result is "SFT high" in Table 1. We see that directly using the weight updates yields lower performance than using activation, and that using activation on the general-purpose calibration achieves the highest overall scores among the three settings. This indicates the need to design protection signals beyond the direct use of weight update.

As an improved version, we then design a mapping function to assign high importance scores to both the smallest and largest weight updates ("SFT both ends"). The reason is that small updates from fine-tuning indicate well-optimized weights even before fine-tuning. To this end, the mapping function takes a quadratic form and is explained in detail in Equation 3. In contrast, we also modify the mapping function of "SFT both ends" to only protect intermediate or small magnitudes, yielding variants we refer to as "SFT mid" and "SFT low", respectively. As shown in Table 1, our "SFT both ends" method yields significantly better scores than others. This empirical improvement validates that $\Delta\mathbf{w}$ strongly correlates to weight importance and protecting both ends of weight updates is effective. The suboptimal performance of protecting intermediate or small magnitudes further indicates that updates on both ends are more important. In Section 5.2, we extend this experiment on more protection signals and other LRMs to demonstrate the effectiveness of our design.

Besides the empirical results, mechanistic interpretability provides strong support for protecting both ends. When plotting our mapped scores of "SFT both ends" on a heatmap, we find significant commonalities between the heatmaps (importance scores of `R1-Distill-Llama-8B` by computing causal relevance of model components to reasoning behaviors) in an existing work (Zhang et al., 2026) and ours. For example, researchers have found that the last-layer MLP up projection is one of the most important components of R1-Llama-8B, and "SFT both ends" also shows this component as the most important. Moreover, the final-layer modules and the MLP gate projection are found overly compressed by existing quantization (*e.g.*, GPTQ and AWQ), and our design precisely protects these LRMs components as shown in Figure 2. Protecting both ends closely matches the causal relationship analysis between reasoning capability and model components in this work, which provides additional interpretability backup to our hypothesis. We provide more related details on the commonalities in Appendix A.3.

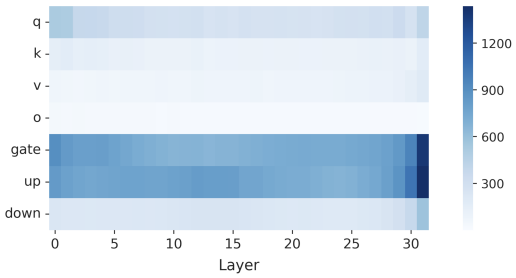


Figure 2: Heatmap of "SFT both ends" on `R1-Distill-Llama-8B`. Our hypothesis of protecting both ends of weight updates closely aligns with existing findings.

3.2 Our Method

To protect both ends, we design the mapping function $f(\cdot)$ as two piecewise quadratic functions, so it assigns higher importance scores to smaller and larger weight updates.

Protecting Both Ends of $\Delta\mathbf{w}$. A quadratic function provides simple and effective mappings that assign weight updates on both ends with high importance scores, which becomes our choice. We also observe that all updates across different modules and layers are unevenly distributed, with many updates clustered at low magnitudes. This suggests to treat the left and right ends of the spectrum

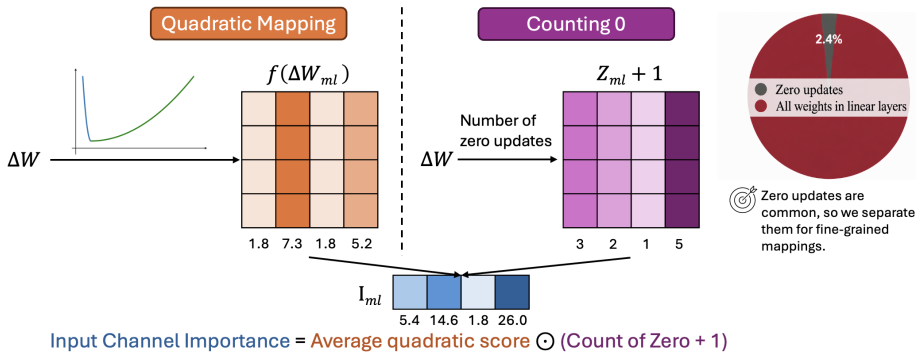


Figure 3: Channel importance $\mathbf{I}_{m\ell}$ of QuantLRM. The input channel importance is the element-wise product of the average quadratic score and the count of zero updates. In practice, we increment the count of zero updates by 1 for each channel as discussed in Section 3.2. Since zero updates are common (*e.g.*, 2.4% of the updates in the fine-tuning traces of `R1-Distill-Qwen-32B` are zero), we handle them separately to enable fine-grained mappings.

separately. We then find the median of all the updates and fit two simple restricted quadratic functions. To be specific, we denote the global minimum update as $\Delta\mathbf{w}_{\text{min}}$, the maximum update as $\Delta\mathbf{w}_{\text{max}}$, the median update as $\Delta\mathbf{w}_{\text{mid}}$, and the output of our mapping function as $\mathbf{y}$. Our mapping function is defined as follows:

$$f(\Delta\mathbf{w}) = \begin{cases} \mathbf{y}_{\text{min}} + (\mathbf{y}_{\text{max}} - \mathbf{y}_{\text{min}}) \left(\dfrac{\Delta\mathbf{w}_{\text{mid}} - \Delta\mathbf{w}}{\Delta\mathbf{w}_{\text{mid}} - \Delta\mathbf{w}_{\text{min}}} \right)^2, & \Delta\mathbf{w} \le \Delta\mathbf{w}_{\text{mid}}, \\ \mathbf{y}_{\text{min}} + (\mathbf{y}_{\text{max}} - \mathbf{y}_{\text{min}}) \left(\dfrac{\Delta\mathbf{w} - \Delta\mathbf{w}_{\text{mid}}}{\Delta\mathbf{w}_{\text{max}} - \Delta\mathbf{w}_{\text{mid}}} \right)^2, & \Delta\mathbf{w} > \Delta\mathbf{w}_{\text{mid}} \end{cases} \tag{3}$$

where we can derive that $f(\Delta\mathbf{w}_{\text{mid}}) = \mathbf{y}_{\text{min}}$ and $f(\Delta\mathbf{w}_{\text{min}}) = f(\Delta\mathbf{w}_{\text{max}}) = \mathbf{y}_{\text{max}}$. As hyper-parameters, we set $\mathbf{y}_{\text{min}} = 1$ and $\mathbf{y}_{\text{max}} = 10$ as the universal values for all experiments in this paper. Experiment on different $\mathbf{y}_{\text{max}}$ values is in Appendix B, showing it is a robust choice. Setting $\mathbf{i}_{cm\ell} = f(\Delta\mathbf{w}_{cm\ell})$, we then apply Equation 2 to get channel importance for protecting both ends. As further illustrated in Appendix A.1, we additionally try piecewise linear mapping and piecewise fourth-order mapping (W-shapped polynomial) to protect weight updates on both ends. Our quadratic mapping is the optimal design across other alternatives, as it is not too complex and not too simple.

Incorporating Zero Weight Updates. We further observe that $\Delta\mathbf{w}_{\text{min}} = 0$ is common for fine-tuned LRMs, as zero weight updates can account for more than 1% of the total. Due to the popularity of zero updates, excluding them from Equation 3 can provide more fine-grained mappings for other updates. Thus, we decide to handle $\Delta\mathbf{w} = 0$ separately and find $\Delta\mathbf{w}_{\text{mid}}$ on positive updates only.

We assign $\mathbf{y}_{\text{min}}$ to zero weight updates since we later introduce mechanisms to highlight them, so the left restricted function has the range $0 < \Delta\mathbf{w} < \Delta\mathbf{w}_{\text{mid}}$. Formally, we modify $f(\Delta\mathbf{w})$ as

$$f(\Delta\mathbf{w}) = \begin{cases} \mathbf{y}_{\text{min}}, & \Delta\mathbf{w} = 0, \\ \mathbf{y}_{\text{min}} + (\mathbf{y}_{\text{max}} - \mathbf{y}_{\text{min}}) \left(\dfrac{\Delta\mathbf{w}_{\text{mid}} - \Delta\mathbf{w}}{\Delta\mathbf{w}_{\text{mid}} - \Delta\mathbf{w}_{\text{min}}} \right)^2, & 0 < \Delta\mathbf{w} \le \Delta\mathbf{w}_{\text{mid}}, \\ \mathbf{y}_{\text{min}} + (\mathbf{y}_{\text{max}} - \mathbf{y}_{\text{min}}) \left(\dfrac{\Delta\mathbf{w} - \Delta\mathbf{w}_{\text{mid}}}{\Delta\mathbf{w}_{\text{max}} - \Delta\mathbf{w}_{\text{mid}}} \right)^2, & \Delta\mathbf{w} > \Delta\mathbf{w}_{\text{mid}}. \end{cases} \tag{4}$$

After assigning $\mathbf{y}_{\text{min}}$ to zero updates, we further add a stronger design to better protect zero updates (since they lie at the far left): counting the number of zero updates within a channel. Counting their occurrences in addition to fitting the two restricted quadratic functions enables us to obtain higher $\mathbf{I}_{cm\ell}$ when a channel contains many weights of zero updates. As a result, we calculate the number

of zero updates within a channel as $\mathbf{Z}_{cm\ell}$, and we compute $\mathbf{I}_{cm\ell}$ based on $f(\Delta\mathbf{w})$ in Equation 4 as

$$\mathbf{I}_{cm\ell} = \left(\frac{1}{N}\sum_{n=1}^{N} f\left(\Delta\mathbf{w}_{cm\ell}^{(n)}\right)\right) \cdot (\mathbf{Z}_{cm\ell} + 1) \tag{5}$$

We add $\mathbf{Z}_{cm\ell}$ by 1 to avoid the cases when $\mathbf{Z}_{cm\ell} = 0$. This approach of protecting both ends and incorporating zero updates is illustrated in Figure 3, with the best overall scores in Table 1. We analyze alternative designs of incorporating zero updates in Appendix A.2. For example, we conduct an experiment on using the ratios of zero updates instead of raw counts. We find that using the current multiplicative form with raw counts is the optimal design across other alternatives.

Injecting Fine-Tuning Signals. After finalizing channel importance in equation 5, we apply it to quantization. We choose to follow AWQ scaling due to its flexibility to incorporate different importance scores. Specifically, scaling factors $\mathbf{s}$ are searched to minimize quantization loss. For input feature $\mathbf{X}$ and fine-tuned weight matrix $\mathbf{W}$, we have the quantization loss:

$$\mathcal{L}(\mathbf{s}) = \left\| Q\big(\mathbf{W} \cdot \mathrm{diag}(\mathbf{s})\big)\Big(\mathrm{diag}(\mathbf{s})^{-1} \cdot \mathbf{X}\Big) - \mathbf{W}\mathbf{X} \right\| \tag{6}$$

To inject our fine-tuning signals, we set:

$$\mathbf{s} = \big(\mathbf{I}_{m\ell}\big)^{\alpha}, \qquad \alpha^* = \arg\min_{\alpha} \mathcal{L}\big(\big(\mathbf{I}_{m\ell}\big)^{\alpha}\big) \tag{7}$$

where $\mathbf{I}_{m\ell}$ is our computed vector consisting of all channel importance scores of a weight matrix. $\mathbf{I}_{m\ell}$ controls the degree of scaling applied to weights, so a more important channel will be scaled up more than a less important one. We search the best α that minimizes the loss function (20 candidates in total) over the interval of [0, 1]. Since $\mathbf{I}_{m\ell}$ is precomputed before the search of α, we go through every target module iteratively to finalize α and save the corresponding $\mathbf{s}$. The actual quantization is then performed by scaling the weights with $\mathbf{s}$ and applying the standard quantization function.

4 Experiment Setting

To verify the effectiveness and applicability of QuantLRM, we evaluate a wide range of fine-tuning strategies and model families, competitive baselines, and reasoning-centric datasets.

Models and Fine-Tuning Pipelines. To show the wide applicability of QuantLRM over different model families and fine-tuning pipelines, we run QuantLRM on SFT, DPO, and RLVR fine-tuned LRMs, from three model families including Llama, Qwen, and Olmo. For SFT, we select DeepSeek-R1 distillation LRMs (Guo et al., 2025), including `R1-Distill-Llama-70B`, `R1-Distill-Qwen-32B`, and `R1-0528-Qwen3-8B`. For DPO, we select `Olmo-3-7B-Think-DPO`. For RLVR, we select `Olmo-3-7B-Think` (Olmo et al., 2025). We leverage the weight updates from `Olmo-3-7B-Think-DPO` to the RLVR version to quantize the RLVR fine-tuned model and the updates from `Olmo-3-7B-Think-SFT` to the DPO version to quantize the DPO model. All these popular LRMs are fine-tuned from base models, and their pre-fine-tuned checkpoints are widely available.

Baselines. We compare QuantLRM against state-of-the-art PTQ methods: GPTQ (Frantar et al., 2022), GPTAQ (Li et al., 2025), AWQ (Lin et al., 2024), ANY3 (Elhoushi & Johnson, 2025), and SliderQuant (Wang et al., 2026). They form a comprehensive list of weight-only quantization methods. We do not report ANY3 on `R1-Llama-70B`, as its 3-bit quantized model is unable to generate coherent text. We run SliderQuant only on `R1-Qwen-32B`, because it does not support newer LRMs such as Olmo3, and running it on `R1-Llama-70B` is prohibitively expensive compared to other methods. For DPO and RLVR fine-tuned LRMs, we report the performance of GPTQ and AWQ due to their overall competitive scores on the SFT models.

Datasets. We select four benchmarks testing a variety of reasoning capabilities, domains, and difficulty levels: AIME-120 for testing mathematical reasoning on 120 complex questions from 2022 to 2025, FOLIO for logical reasoning, temporal sequences of BIG-Bench Hard for temporal reasoning, and GPQA-Diamond for reasoning across multiple scientific domains. We measure accuracy of LRMs on each benchmark and report the average score. We use a combination of AIME 2021 (Veeraboina, 2023) and 100 random instances from the FOLIO training set as validation data.

Table 2: Main results. QuantLRM delivers consistent improvements over all baselines across various LRMs on 3-bit quantization. The highest scores of each model are **bold** excluding 16-bit. Blue rows of "QuantLRM" indicate our results. Gray rows of "16-bit original" indicate upper bound.

Model Variants			Accuracy				
Model	Type	Precision	AIME-120	FOLIO	Temporal	GPQA-Diamond	Avg
R1-Llama-70B	SFT	3-bit GPTQ	37.5	77.3	99.2	**60.6**	68.65
R1-Llama-70B	SFT	3-bit GPTAQ	38.3	76.8	99.2	50.0	66.08
R1-Llama-70B	SFT	3-bit AWQ	44.2	74.9	**100.0**	57.6	69.18
R1-Llama-70B	SFT	3-bit QuantLRM	**49.2**	**78.3**	99.6	58.1	**71.30**
R1-Llama-70B	SFT	16-bit original	56.7	79.8	100.0	60.6	74.28
R1-Qwen-32B	SFT	3-bit GPTQ	41.7	77.3	99.6	54.0	68.15
R1-Qwen-32B	SFT	3-bit GPTAQ	35.0	72.9	99.6	60.1	66.90
R1-Qwen-32B	SFT	3-bit AWQ	50.0	77.3	99.6	55.1	70.50
R1-Qwen-32B	SFT	3-bit ANY3	44.2	80.3	99.6	58.1	70.55
R1-Qwen-32B	SFT	3-bit SliderQuant	33.3	76.8	**100.0**	58.6	67.18
R1-Qwen-32B	SFT	3-bit QuantLRM	**53.3**	**80.8**	99.6	**60.6**	**73.58**
R1-Qwen-32B	SFT	16-bit original	56.7	83.7	99.2	61.1	75.18
R1-Qwen3-8B	SFT	3-bit GPTQ	8.3	77.3	82.8	35.9	51.08
R1-Qwen3-8B	SFT	3-bit GPTAQ	6.7	**78.8**	84.4	30.3	50.05
R1-Qwen3-8B	SFT	3-bit AWQ	34.2	77.8	98.4	47.5	64.48
R1-Qwen3-8B	SFT	3-bit ANY3	15.0	69.0	54.4	30.3	42.18
R1-Qwen3-8B	SFT	3-bit QuantLRM	**40.0**	76.8	**99.2**	**48.5**	**66.13**
R1-Qwen3-8B	SFT	16-bit original	59.2	79.8	99.6	63.6	75.55
Olmo-3-7B-Think-DPO	DPO	3-bit GPTQ	16.7	63.5	95.6	**39.4**	53.80
Olmo-3-7B-Think-DPO	DPO	3-bit AWQ	32.5	**69.5**	94.8	35.9	58.18
Olmo-3-7B-Think-DPO	DPO	3-bit QuantLRM	**35.8**	65.0	**96.4**	36.4	**58.40**
Olmo-3-7B-Think-DPO	DPO	16-bit original	57.5	70.4	98.8	49.0	68.93
Olmo-3-7B-Think	RLVR	3-bit GPTQ	21.7	69.5	93.6	41.9	56.68
Olmo-3-7B-Think	RLVR	3-bit AWQ	28.3	**76.4**	84.0	35.9	56.15
Olmo-3-7B-Think	RLVR	3-bit QuantLRM	**38.3**	74.4	**96.8**	**43.4**	**63.23**
Olmo-3-7B-Think	RLVR	16-bit original	54.2	75.4	99.6	45.5	68.68

We follow each baseline's default calibration data. QuantLRM uses AWQ's smaller calibration set, while our reasoning calibration uses verified reasoning chains (Guha et al., 2025) at the same size as other baselines. Since larger reasoning calibration does not improve activation-based quantization (*e.g.*, Table 1), we retain the default setups for fair comparison. More details are in Appendix F.

5 Results and Analysis

5.1 Overall Results

Our overall results of 3-bit quantization are in Table 2, demonstrating that QuantLRM yields state-of-the-art performance on various reasoning tasks compared to all the competitive baselines. Since 4-bit quantization is mostly saturated on reasoning benchmarks across different methods, we present experiments on W4A16 in Appendix D. The target of QuantLRM is to push the boundary towards sub-4-bit quantization, which is a more challenging scenario with more room for improvement.

Improvement on SFT. QuantLRM delivers consistent improvements over GPTQ, GPTAQ, AWQ, ANY3, and SliderQuant across all three SFT models, increasing the best average scores from baselines by up to 3.03%. QuantLRM improves the average 3-bit score over the strongest PTQ baseline by 2.12% on R1-Llama-70B. Even on R1-Qwen3-8B, which has far fewer parameters (and thus less room for improvement), QuantLRM improves the average score by 1.65% over the best baseline. Since AIME-120 is the hardest task selected (largest performance drop after quantization), QuantLRM demonstrates its advantage by consistently leading on this benchmark by a significant margin. For R1-Qwen-32B, SliderQuant leads on temporal reasoning but yields lower scores on other benchmarks. This is expected due to its focus on basic language generation and simple reasoning tasks, since temporal reasoning is the easiest of the selected tasks as reflected by consistently high scores across methods. The strongest performance of QuantLRM demonstrates the advantage of using SFT signals for quantization.

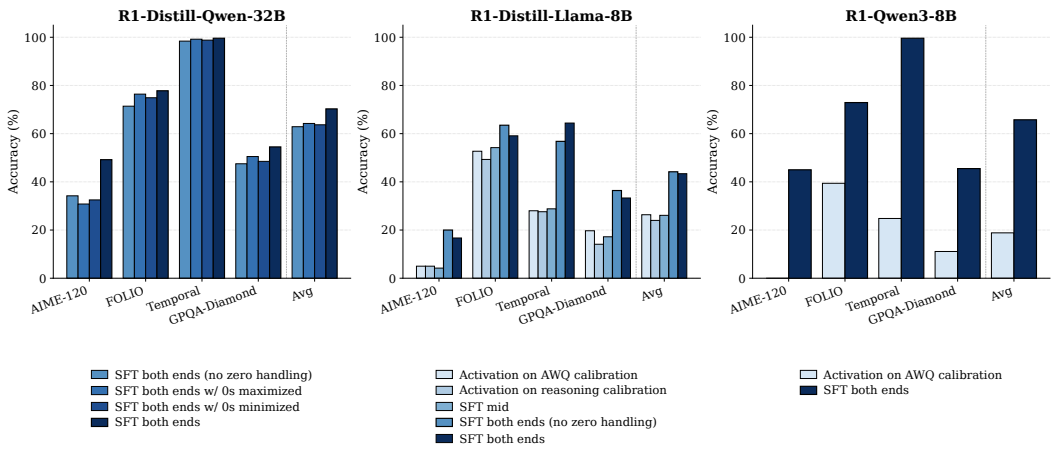


Figure 4: Ablation study. Mixed-bit quantization performance of three LRMs when quantized to 3-bit while keeping a fraction of channels in 16-bit, selected by different protection signals. Across all three models, *SFT both ends* (darkest blue) yields the highest overall accuracy.

Improvement on DPO and RL. QuantLRM also delivers consistent improvements over the most competitive baselines on DPO and RL fine-tuned LRMs, increasing the average score by up to 6.55% over the best baselines. Similar to SFT models, QuantLRM leads on AIME-120. The performance gain demonstrates that QuantLRM works on various types of fine-tuning. The fundamental reason is that the traces left by different fine-tuning strategies overlap significantly. The gain from QuantLRM is smallest on DPO, which we attribute to relatively weak fine-tuning signals. Based on Olmo3 report (Olmo et al., 2025), the performance gain from the DPO variant over SFT is much smaller than that from RLVR over DPO. Therefore, the corresponding fine-tuning signals we can collect from DPO are also weaker than RLVR (we provide more details on fine-tuning strength in Appendix H). The alignment between quantization and fine-tuning performance gains further demonstrates our core idea: fine-tuning traces can guide quantization decisions.

Significance Tests. To test statistical significance, we conduct paired-bootstrap tests comparing 3-bit QuantLRM with AWQ on `Olmo-3-7B-Think-DPO` and `R1-Qwen3-8B`, where AWQ is the strongest baseline and our performance gains are smallest. If our gains on these models are statistically significant, the gains on the other models are likely even stronger. Using 200000 bootstrap repetitions with 50000 paired draws per benchmark and equal weights across four benchmarks, QuantLRM improves macro accuracy by 0.251 and 1.665 percentage points, respectively, with corresponding 95% bootstrap confidence intervals of [+0.047, +0.455] and [+1.476, +1.854], indicating statistically significant gains under this sampling configuration. More details are in Appendix C.

5.2 Ablation Study

Studying each component in QuantLRM, we conduct ablation analysis on more LRMs in Figure 4 using the same mixed-precision setting as in Section 3.1. Scores on `R1-Qwen-32B` highlight the value of explicitly handling zero updates. When protecting both ends without handling zero updates separately, we refer to Equation 3, which yields the lowest average score. Then, instead of counting zero updates, we choose to directly assign them to $\mathbf{y}_{\text{max}}$ or $\mathbf{y}_{\text{min}}$. Both choices provide modest gains, especially when zero updates are maximized. Our best design counts zero updates and excludes them when fitting the quadratic functions, resulting in a 5.95% improvement in average score.

On the two 8B LRMs, "SFT both ends" yields much better scores than activation-based signals. On `R1-Llama-8B`, protecting intermediate magnitudes ("SFT mid") is not helpful, which echoes our discussion in 3.1 and reinforces our motivation for protecting both ends. Notably, on this model, protecting both ends without separately handling zero updates achieves the best average score, slightly outperforming counting zero updates. This contrast highlights the diversity of our LRMs: counting zero updates is the best design on the 32B model, and since it is only marginally worse on

Table 3: Time cost and inference speed. QuantLRM adds little overhead for preparation and search, while achieving the same latency as AWQ.

Model	Method	Prep Time	Search Time	Tokens/s
`Olmo-3-7B-Think`	4-bit AWQ	-	6min28s	30.24
`Olmo-3-7B-Think`	4-bit QuantLRM	2min27s	6min35s	30.33

Table 4: Performance comparison of different protected regions in actual deployment setting.

Model	Method	AIME-120	FOLIO	Temporal	GPQA-Diamond	Avg
`R1-Qwen-32B`	Protect lows	43.3	79.3	**99.6**	**61.1**	70.83
`R1-Qwen-32B`	Protect middle	46.7	79.3	**99.6**	56.6	70.55
`R1-Qwen-32B`	Protect highs	49.2	78.3	99.2	60.1	71.70
`R1-Qwen-32B`	Protect both ends (QuantLRM)	**53.3**	**80.8**	**99.6**	60.6	**73.58**

`R1-Llama-8B`, we adopt counting zero updates as our final design. This ablation demonstrates the benefits of protecting both ends and using the zero-update count to amplify channel importance.

5.3 Cost of Time and Inference Speed

QuantLRM requires computing weight updates, mapping them, and counting zero updates as preparation prior to quantization. During quantization, the core is to search for optimal scaling factors, so we record the search time. We measure inference latency by reporting the median latency over seven prompts (Appendix G). This experiment is conducted on 4-bit quantized `Olmo-3-7B-Think`.

In Table 3, we observe that QuantLRM incurs a small one-time preparation overhead of only 2 minutes and 27 seconds, which is not present in AWQ. Importantly, this extra cost is *offline* (prior to quantization) and does not affect inference. The upper bound of preparation time is from `R1-Llama-70B`, which takes 87 minutes and 22 seconds. Given that a 70B-LRM is a large model for recent quantization analysis (Zhang et al., 2026; Liu et al., 2025), this offline preparation time showcases high feasibility. During quantization, in Table 3, both methods take roughly the same amount of time, with QuantLRM spending a few extra seconds loading the precomputed fine-tuning signals. Their inference latency is the same, since we use the same AWQ kernel for inference. Besides the time cost, QuantLRM consumes similar amount of GPU memory as AWQ as well. Therefore, paying a small amount of extra cost mainly during preparation, QuantLRM is a reasonably efficient method that delivers significantly better reasoning accuracy after quantization.

5.4 Shifting Protected Regions in Actual Deployment Setting

To show the alignment between mixed-precision setting and actual quantization deployment (AWQ scaling), we present the scores in the actual deployment that shift the protected regions in Table 4. Here are our concrete settings: protecting lows (left quadratic piece with zero handling, assigning all weights on the right to $\mathbf{y}_{\min}$), protecting middle (reflecting our piecewise quadratic mapping without zero handling), protecting highs (right quadratic piece without zero handling, assigning all weights on the left to $\mathbf{y}_{\min}$), and protecting both ends (our piecewise quadratic mapping with zero handling).

First, protecting both ends yields the best scores, which is the optimal mapping. Note that all other protected regions are showing better (or equal) average scores than baselines on `R1-Qwen-32B`, which demonstrates the utility of fine-tuning signals. Second, moving from mixed-precision to AWQ scaling, we see our findings generalize to actual deployment (*e.g.*, protecting both ends always yields the best scores). This shows that our mixed-precision experiments validate our hypothesis in 3.1.

6 Conclusion

We propose QuantLRM for quantization of LRMs via fine-tuning signals. Delivering state-of-the-art performance, QuantLRM assigns high importance scores to protect the smallest and largest weight updates. More broadly, our findings show that fine-tuning traces provide strong and lightweight signals for PTQ, offering a practical direction to future research.

AI USE STATEMENT

In this work, we used generative AI tools for implementing methods. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, and interpreting results. Generating synthetic data sets, assisting in the writing of proofs, assisting with translation, cleaning and reformating dataset, and supporting qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools for creating or modifying scientific figures or images, creating or editing software code, and editing a research paper to improve readability. We have reviewed all AI-assisted work. For example, we checked all LLM-generated code to ensure its correctness and efficiency to the highest possible standard. We also avoided prompting LLMs to generate paragraphs or entire sections; instead, we only asked LLMs to modify existing individual sentences. We proofread all LLM-generated sentences for accuracy and coherency. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

As clearly stated in the footnote of our abstract, the code for our method will be released upon paper acceptance. For reproducibility during the review stage, we have also uploaded an anonymized version of our code, together with a README file, as supplementary material. QuantLRM is designed as an open-source method to facilitate LLM compression and AI research more broadly, similar to the baseline methods used in this paper.

REFERENCES

Ju-Seung Byun, Jiyun Chun, Jihyung Kil, and Andrew Perrault. ARES: Alternating reinforcement learning and supervised fine-tuning for enhanced multi-modal chain-of-thought reasoning through diverse AI feedback. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 4410–4430, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.252. URL https://aclanthology.org/2024.emnlp-main.252/.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. URL https://arxiv.org/abs/1810.04805.

Hongzhe Du, Weikai Li, Min Cai, Karim Saraipour, Zimin Zhang, Himabindu Lakkaraju, Yizhou Sun, and Shichang Zhang. How post-training reshapes llms: A mechanistic view on knowledge, truthfulness, refusal, and confidence, 2025. URL https://arxiv.org/abs/2504.02904.

Sarah Egler, John Schulman, and Nicholas Carlini. Detecting adversarial fine-tuning with auditing agents, 2025. URL https://arxiv.org/abs/2510.16255.

Mostafa Elhoushi and Jeff Johnson. any4: Learned 4-bit numeric representation for llms, 2025. URL https://arxiv.org/abs/2507.04610.

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.

Avichal Goel, Yoon Kim, Nir Shavit, and Tony T. Wang. Learning to interpret weight differences in language models, 2025. URL https://arxiv.org/abs/2510.05092.

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.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, Wanjia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik Sharma, Charlie Cheng-Jie Ji, Yichuan Deng, Sarah Pratt, Vivek Ramanujan, Jon Saad-Falcon, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal Shankar, Aaron Gokaslan, Mike A. Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alexandros G. Dimakis, and Ludwig Schmidt. Openthoughts: Data recipes for reasoning models, 2025. URL https://arxiv.org/abs/2506.04178.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645 (8081):633–638, September 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alexander Wardle-Solano, Hannah Szabó, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, Ansong Ni, Linyong Nan, Jungo Kasai, Tao Yu, Rui Zhang, Alexander Fabbri, Wojciech Maciej Kryscinski, Semih Yavuz, Ye Liu, Xi Victoria Lin, Shafiq Joty, Yingbo Zhou, Caiming Xiong, Rex Ying, Arman Cohan, and Dragomir Radev. FOLIO: Natural language reasoning with first-order logic. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 22017–22031, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1229. URL https://aclanthology.org/2024.emnlp-main.1229/.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Yuhang Li, Ruokai Yin, Donghyun Lee, Shiting Xiao, and Priyadarshini Panda. GPTAQ: Efficient finetuning-free quantization for asymmetric calibration. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=QdELyl0FST.

Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. Zebralogic: On the scaling limits of llms for logical reasoning, 2025. URL https://arxiv.org/abs/2502.01100.

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

Ruikang Liu, Yuxuan Sun, Manyi Zhang, Haoli Bai, Xianzhi Yu, Tiezheng Yu, Chun Yuan, and Lu Hou. Quantization hurts reasoning? an empirical study on quantized reasoning models. *arXiv preprint arXiv:2504.04823*, 2025.

Samuel Marks, Johannes Treutlein, Trenton Bricken, Jack Lindsey, Jonathan Marcus, Siddharth Mishra-Sharma, Daniel Ziegler, Emmanuel Ameisen, Joshua Batson, Tim Belonax, Samuel R. Bowman, Shan Carter, Brian Chen, Hoagy Cunningham, Carson Denison, Florian Dietz, Satvik Golechha, Akbir Khan, Jan Kirchner, Jan Leike, Austin Meek, Kei Nishimura-Gasparian, Euan Ong, Christopher Olah, Adam Pearce, Fabien Roger, Jeanne Salle, Andy Shih, Meg Tong, Drake Thomas, Kelley Rivoire, Adam Jermyn, Monte MacDiarmid, Tom Henighan, and Evan Hubinger. Auditing language models for hidden objectives, 2025. URL https://arxiv.org/abs/2503.10965.

Mathematical Association of America. MAA Invitational Competitions. URL https://maa.org/maa-invitational-competitions/. Accessed: 2025-03-19.

Team Olmo, :, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, Pradeep Dasigi, Robert Berry, Saumya Malik, Saurabh Shah, Scott Geng, Shane Arora, Shashank Gupta, Taira Anderson, Teng Xiao, Tyler Murray, Tyler Romero, Victoria Graf, Akari Asai, Akshita Bhagia, Alexander Wettig, Alisa Liu, Aman Rangapur, Chloe Anastasiades, Costa Huang, Dustin Schwenk, Harsh Trivedi, Ian Magnusson, Jaron Lochner, Jiacheng Liu, Lester James V. Miranda, Maarten Sap, Malia Morgan, Michael Schmitz, Michal Guerquin, Michael Wilson, Regan Huff, Ronan Le Bras, Rui Xin, Rulin Shao, Sam Skjonsberg, Shannon Zejiang Shen, Shuyue Stella Li, Tucker Wilde, Valentina Pyatkin, Will Merrill, Yapei Chang, Yuling Gu, Zhiyuan Zeng, Ashish Sabharwal, Luke Zettlemoyer, Pang Wei Koh, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. Olmo 3, 2025. URL https://arxiv.org/abs/2512.13961.

Cuong Pham, Hoang Anh Dung, Cuong C. Nguyen, Trung Le, Gustavo Carneiro, and Thanh-Toan Do. Layer-wise high-impact parameter ratio optimization in post-training quantization for large language models, 2025. URL https://arxiv.org/abs/2511.17801.

Bowen Ping, Shuo Wang, Hanqing Wang, Xu Han, Yuzhuang Xu, Yukun Yan, Yun Chen, Baobao Chang, Zhiyuan Liu, and Maosong Sun. Delta-come: Training-free delta-compression with mixed-precision for large language models, 2024. URL https://arxiv.org/abs/2406.08903.

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, 2023. URL https://arxiv.org/abs/1910.10683.

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

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them, 2022. URL https://arxiv.org/abs/2210.09261.

Hemish Veeraboina. Aime problem set 1983-2024, 2023. URL https://www.kaggle.com/datasets/hemishveeraboina/aime-problem-set-1983-2024.

Shigeng Wang, Chao Li, Yangyuxuan Kang, Jiawei Fan, Zhonghong Ou, and Anbang Yao. Sliderquant: Accurate post-training quantization for llms, 2026. URL https://arxiv.org/abs/2603.25284.

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

Xiaoming Yu, Shize Tang, Guanghua Yu, Linchuan Xie, Song Liu, Jianchen Zhu, and Feng Li. Daq: Delta-aware quantization for post-training llm weight compression, 2026. URL https://arxiv.org/abs/2603.22324.

Nan Zhang, Eugene Kwek, Yusen Zhang, Ngoc-Hieu Nguyen, Prasenjit Mitra, and Rui Zhang. When reasoning meets compression: Understanding the effects of llms compression on large reasoning models, 2026. URL https://arxiv.org/abs/2504.02010.

Zhong Zhang, Bang Liu, and Junming Shao. Fine-tuning happens in tiny subspaces: Exploring intrinsic task-specific subspaces of pre-trained language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1701–1713, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.95. URL https://aclanthology.org/2023.acl-long.95/.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. Agieval: A human-centric benchmark for evaluating foundation models, 2023. URL https://arxiv.org/abs/2304.06364.

A JUSTIFICATION OF OUR WEIGHT IMPORTANCE SCORE FUNCTIONS

A.1 ALTERNATIVE MAPPING FUNCTIONS

The observed gains are rooted in fine-tuning traces, whose potential is further realized through our quadratic mapping. Exploring alternative mappings, we additionally try piecewise linear mapping and piecewise fourth-order mapping (W-shapped polynomial) to protect weight updates on both ends. Both mapping functions adopt our zero handling to allow a fair comparison against our "SFT both ends" in Table 1. We use the same setting as Table 1 and report the scores in Table 5.

"SFT both ends" is the optimal design across other alternatives with at least 1.35% improvement on the average score. Compared against the two rows using activation signals in Table 1, both linear and fourth-order mappings yield better performance on the average score, which indicates the effectiveness of protecting both ends. Since the fourth-order mapping is W-shapped, it also assigns higher importance scores to the weights with intermediate-magnitude updates. Our quadratic mapping is the optimal design across other alternatives, as it is not too complex and not too simple. Since most weight-update magnitudes are tiny and their distribution remains uneven even within the regions on either side of the median, our quadratic mapping avoids both underfitting (with linear mapping) and overfitting (with higher-order polynomials) in machine learning.

A.2 ALTERNATIVE FORMS OF INCORPORATING ZERO WEIGHT UPDATES

The multiplicative form of Equation 5 is the most straightforward and effective approach we find to incorporate zero weight updates and quadratic values. By simply taking their product, we use the number of zero updates as an amplifier without allowing it to dominate the average quadratic values in Equation 5. As they represent different meanings and zero updates are highly prevalent as shown in Figure 3, computing an additive form will inevitably downplay the average quadratic values (the number of zero updates is typically a much higher number), while ratios or normalized statistics of zero updates may underestimate their popularity. For example, we show the performance of a ratio-based method for incorporating zero updates using mixed-precision quantization (the same setting as Table 1) in Table 6. Replacing raw counts, we instead use the percentages of zero updates within channels.

The ratio-based method is more than 2% lower than our raw counts in terms of average accuracy. Referring to Table 1, it is also slightly lower than using activation with general-purpose calibration data. The reason is that such ratios significantly reduce the magnitudes of the raw counts, which essentially understates the prevalence of zero updates. Since our goal is to handle zero updates separately and provide more fine-grained quadratic mappings for other updates, we believe the current multiplicative form is the optimal design across other alternatives.

Table 5: Performance of alternative mapping functions under the mixed-precision setting. Our quadratic mapping yields the best overall performance, which is the optimal design across other alternatives.

		Accuracy				
Model	Mapping Function	AIME-120	FOLIO	Temporal	GPQA-Diamond	Avg
`R1-Distill-Qwen-32B`	Linear	42.5	74.9	99.2	**59.1**	68.93
`R1-Distill-Qwen-32B`	Fourth-order polynomial	45.8	76.8	**100.0**	51.5	68.53
`R1-Distill-Qwen-32B`	Quadratic	**49.2**	**77.8**	99.6	54.5	**70.28**

Table 6: Performance comparison of different methods to incorporate zero-update weights.

Model	Method	AIME-120	FOLIO	Temporal	GPQA-Diamond	Avg
`R1-Qwen-32B`	SFT both ends (ratio-based)	47.5	**77.8**	98.4	49.0	68.18
`R1-Qwen-32B`	Protect both ends (raw counts)	**49.2**	**77.8**	**99.6**	**54.5**	**70.28**

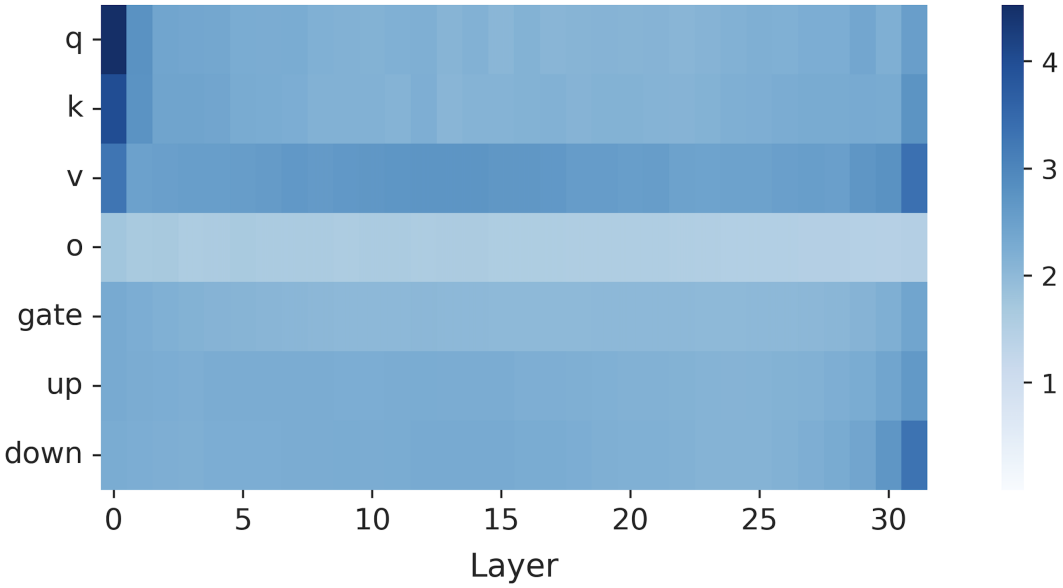


Figure 5: Heatmap of "SFT both ends" without zero handling on `R1-Distill-Llama-8B`. Protecting both ends while relaxing the zero-handling mechanism shows some findings similar to those of existing work, but still presents major differences.

A.3 Interpretability of Protecting Both Ends

We demonstrate the necessity of protecting both ends with zero handling from the mechanistic interpretation perspective. We refer to the top four heatmaps in Figure 2 of a recent work on LRMs compression (Zhang et al., 2026), which represent the importance scores of `DeepSeek-R1-Distill-Llama-8B` by computing causal relevance to reasoning behaviors. We show our heatmap of "SFT both ends" in Figure 2. There are many commonalities between our heatmap and those in the recent work. In the LRMs compression work, one of the most important findings is that final-layer up projection is the most important component, which is also the case in our heatmap. Researchers find that the last-layer gate projection appears to be the second most important component in their heatmap, while the same component is also assigned the second highest importance score in Figure 2. Therefore, besides the strong empirical results shown in our paper, mechanistic interpretability provides strong support for protecting both ends.

We also show that removing a specific component of our method moves us farther away from the existing findings. For instance, we show the heatmap of relaxing zero-handling mechanism in Figure 5. Comparing the heatmaps of having or removing zero handling, we see that removing the zero handling will lead to relatively low importance scores to the final-layer up and gate projections. The final up projection receives the highest importance score and the final gate projection receives the second highest only when we add zero handling back, exactly matching the current findings. However, removing zero handling does reveal some similarities to existing findings, thanks to protecting other weight updates at the low and high ends, such as relatively low importance scores for the self-attention output projection across all layers. Still, this is insufficient to match the heatmaps reported in prior work.

B Implementation Details

During model decoding, we use vLLM for inference optimization and adopt the same set of hyperparameters across all methods (including the 16-bit original LRM). Aligning with the DeepSeek report (Guo et al., 2025), we set maximum generation length to 32768, temperature to 0.6, and top-p value to 0.95. Our consistent decoding mechanism allows a fair comparison among all methods.

Table 7: Mixed-precision quantization performance based on different $\mathbf{y}_{\text{max}}$ values. We follow the same settings as Table 1.

$\mathbf{y}_{\text{max}}$	AIME-120	FOLIO	Temporal	GPQA-Diamond	Avg
10	49.2	77.8	99.6	54.5	70.28
5	40.0	75.9	99.6	53.5	67.25
15	46.7	73.4	99.2	55.1	68.60

For all experiments, we set $\mathbf{y}_{\text{min}} = 1$ and $\mathbf{y}_{\text{max}} = 10$. Since our pipeline uses only 16-bit arithmetic, large models can produce large importance scores (*e.g.*, large matrices often yield more zero updates), leading to overflow; we mitigate this by slicing $\Delta\mathbf{W}$ of MLP modules into pieces (*e.g.*, 2 or 5 pieces) when counting zero updates (used for `R1-Llama-70B` and `R1-Qwen-32B`). We then multiply the average zero-update count by the quadratic mapping scores in Equation 5 to obtain the final channel importance. We mainly focus on analyzing 3-bit quantization (W3A16) performance and also show 4-bit performance in Appendix D, since 4-bit PTQ baselines have almost lossless scores (Zhang et al., 2026) and our priority is on 3-bit. When reporting accuracy scores of all methods, we use vLLM (with access to A100 80GB GPUs) to perform inference on pseudo-quantized LRMs. In Section 5.3, we first perform 4-bit quantization and then use the AWQ kernel to measure inference latency, since QuantLRM is compatible with AWQ kernel and the kernel supports only 4-bit LRMs.

Experiment on Different Hyperparameters. $\mathbf{y}_{\text{min}} = 1$ and $\mathbf{y}_{\text{max}} = 10$ are the universal choices across all the experiments in the main content. With $\mathbf{y}_{\text{min}} = 1$ all the time, we try two more $\mathbf{y}_{\text{max}}$ values (5 and 15), since the goal is to study the range between $\mathbf{y}_{\text{min}}$ and $\mathbf{y}_{\text{max}}$. We provide the performance scores of all three $\mathbf{y}_{\text{max}}$ values in Table 7 (the same settings as Table 1 except different $\mathbf{y}_{\text{max}}$ values).

We observe $\mathbf{y}_{\text{max}} = 10$ yields the best scores, indicating that performance can vary based on a wider or narrower range. With state-of-the-art quantization performance displayed in the main content, QuantLRM uses the universal $\mathbf{y}_{\text{min}}$ and $\mathbf{y}_{\text{max}}$ values for all the models, which demonstrates their wide applicability and robustness across different model families. A more fine-grained grid search shall be left to future engineering works.

C Significance Tests

To demonstrate the robustness of our results, we conduct bootstrap-based significance tests using the outputs of 3-bit QuantLRM and 3-bit AWQ on `Olmo-3-7B-Think-DPO` and `R1-Qwen3-8B`. We select these two models because QuantLRM has its smallest performance margin over AWQ on them, while AWQ is the strongest baseline for both models. If our performance gains on these models are statistically significant, the gains on the other models are likely to be even stronger.

In our bootstrap sampling, each bootstrap repetition samples 50000 paired observations with replacement from each of our four benchmarks, producing 200000 total draws. It performs 200000 bootstrap repetitions with seed 1 and an automatically selected computational batch size of 20. We choose a high number of total draws in order to test statistical stability. The accuracy difference between QuantLRM and AWQ is calculated separately for each benchmark and combined using equal 25% benchmark weights.

QuantLRM achieves statistically better accuracy than AWQ on both LRMs. On `Olmo-3-7B-Think-DPO`, QuantLRM achieves an equal-weight macro accuracy of 58.405%, versus 58.154% for AWQ, yielding an advantage of 0.251 percentage points. The paired-bootstrap standard error - calculated as the standard deviation of the bootstrapped accuracy differences - is 0.104 percentage points. Under this bootstrap setting, QuantLRM significantly outperforms AWQ (95% bootstrap confidence interval: [+0.047, +0.455]). On `R1-Qwen3-8B`, QuantLRM achieves an equal-weight macro accuracy of 66.133%, versus 64.469% for AWQ, yielding an advantage of 1.665 percentage points. The standard deviation of the bootstrapped accuracy differences is 0.096 percentage points. Under this bootstrap setting, QuantLRM significantly outperforms AWQ (95% bootstrap confidence interval: [+1.476, +1.854]).

Table 8: Performance of 4-bit quantization on `R1-Llama-70B`. At 4 bits, performance is saturated, with all methods matching or exceeding the original 16-bit model.

Model Variants			Accuracy				
Model	Type	Precision	AIME-120	FOLIO	Temporal	GPQA-Diamond	Avg
`R1-Llama-70B`	SFT	16-bit	56.7	79.8	100.0	60.6	74.28
`R1-Llama-70B`	SFT	4-bit GPTQ	55.8	79.3	98.8	64.6	74.63
`R1-Llama-70B`	SFT	4-bit AWQ	60.8	77.8	99.2	64.6	75.60
`R1-Llama-70B`	SFT	4-bit QuantLRM	58.3	77.3	99.6	66.2	75.35

Our reasoning benchmarks such as GPQA-Diamond are generally smaller with high-quality and challenging questions. These benchmarks are popular ones for measuring LRMs' capabilities. We have also included several years of AIME to form AIME-120 in order to have more test data.

D Performance of W4A16 Quantization

We report W4A16 performance in Table 8. The average scores of all three methods surpass the 16-bit LRM, which echoes recent analyses (Zhang et al., 2026; Liu et al., 2025). Even on AIME-120 and GPQA-Diamond (challenging benchmarks), the 16-bit model ranks third or worse. This is the reason why we primarily focus on the evaluation of W3A16, as 4-bit LRMs perform similarly with little room of improvement. As discussed in Section 5.1, QuantLRM pushes the boundary of sub-4-bit quantization. From a practical perspective, these findings indicate that QuantLRM serves as a "safe default" across bit-widths: it provides substantial improvements in W3A16 while remaining competitive and stable in W4A16, where current PTQ methods already saturate in performance.

E Pseudo-Fine-Tuning

QuantLRM works with any post-trained open-source LRMs. In extremely rare situations where a pre-fine-tuned checkpoint of an LRM is unavailable, we show that researchers can gather fine-tuning signals themselves via pseudo-fine-tuning. Selecting `Qwen3-1.7B` as the post-trained LRM and performing SFT on the `OpenR1-Math-94k` dataset from LlamaFactory (Zheng et al., 2024), we find that the performance of QuantLRM steadily increases and starts to surpass AWQ at the 2000th training step, which is still early and far from convergence. The increasingly strong fine-tuning signals improve quantization performance, and the alignment between quantization performance and training progress echoes the utility of fine-tuning signals. It is worth noting that `Qwen3-1.7B` has its pre-fine-tuned version available, and we assume its unavailability to study the effectiveness of pseudo-fine-tuning.

The optimal number of steps depends less on the model and more on the dataset size and training stage, so our protocol is to conduct SFT on the same dataset as ours with batch size of 16, gradient accumulation steps of 4, learning rate (cosine learning rate scheduler) of 2.0e-5, and training epochs of 5. For the purpose of pseudo-fine-tuning, we only recommend SFT due to its better stability than RL (Byun et al., 2024). We recommend saving the latest 10 checkpoints at intervals of 200 training steps to form a comprehensive pool of candidates for quantization.

Because the ultimate goal is to quantize LRMs, fine-tuning can be stopped at any point once quantization performance is satisfactory. We show the average accuracy (on the four selected benchmarks) of AWQ and QuantLRM that gathers fine-tuning signals at different training steps in Table 9. We observe that QuantLRM peaks at the 2400th step. After that, its performance starts to decline. Except the 3800th step, QuantLRM surpasses AWQ at all other steps. As mentioned previously in our protocol, we recommend saving checkpoints at various steps to form a candidate pool, as pseudo-fine-tuning allows us to access intermediate checkpoints. We find that our protocol works for different LRMs, since we adopt a large fine-tuning dataset (94k instances) with key hyperparameters specified.

Table 9: Average accuracy on the four selected benchmarks of AWQ and QuantLRM. QuantLRM gathers fine-tuning signals at different training steps from 2000 to 3800. Except the 3800th step, QuantLRM surpasses AWQ at all other steps.

AWQ	2000	2400	3000	3600	3800
22.33	22.38	27.20	24.23	23.75	20.58

F A Smaller Calibration Set of QuantLRM and AWQ

We use the default calibration size for all quantization methods. For GPTQ, GPTAQ, ANY3, and SliderQuant, we use 128 samples of length 2048 tokens from the C4 dataset (Raffel et al., 2023). For AWQ and QuantLRM, we use 128 samples of length 512 instead. Note that QuantLRM still outperforms other baselines despite using one of the smallest calibration sets, demonstrating the efficacy of our method.

G Prompts Used for Speed Analysis

We use 7 short prompts to measure the inference speed of QuantLRM and AWQ:

- "*Deeply analyze; then decide.*"
- "*Enumerate approaches; pick best.*"
- "*Solve; stress-test; revise.*"
- "*Assume nothing; derive everything.*"
- "*Answer; justify; quantify uncertainty.*"
- "*Decompose; solve parts; recombine.*"
- "*List risks; add safeguards; finalize.*"

The goal is to generate 200 reasoning tokens for each prompt and compute the median latency.

H Fine-Tuning Strength Affects Quantization Performance

The effectiveness of fine-tuning affects the performance gain of QuantLRM. Note that `Olmo-3-7B-Think-DPO` delivers a relatively small performance gain, so we get weaker fine-tuning signals than the RLVR pipeline. Referring to the Olmo-3 report (Olmo et al., 2025), we copy some of the evaluation scores of `Olmo-3-7B-Think` variants in Table 10. The training order is SFT, followed by DPO, and then RLVR.

In general, we can see that the performance gain from the DPO variant over SFT is much smaller than that from RLVR over DPO. Therefore, the corresponding fine-tuning signals we can collect from DPO are also weaker than RLVR. The alignment between quantization and fine-tuning performance gains further demonstrates our core idea: fine-tuning traces offer priceless information for quantization, no matter whether they belong to DPO, SFT, or RLVR.

Table 10: Performance scores of three Olmo-3 variants on BBH (Suzgun et al., 2022), ZebraLogic (Lin et al., 2025), and AGI Eval (Zhong et al., 2023). The performance gain from the DPO variant over SFT is much smaller than that from RLVR over DPO.

Model	BBH	ZebraLogic	AGI Eval	Average
`Olmo-3-SFT`	84.1	57.9	77.2	73.07
`Olmo-3-DPO`	83.7	60.6	79.1	74.47
`Olmo-3-RLVR`	86.6	66.5	81.5	78.20