Does Quantization Affect Multilingual Reasoning Behavior?

Anonymous ACL submission

Abstract

Though quantization is widely used to reduce the inference and serving costs of large language models, its impact on multilingual reasoning remains poorly understood. Multilingual users require both final-answer correctness and native language consistency in the response; making it critical to understand the effect of quantization in the reasoning era. We study quantized multilingual reasoning models under 7 quantization methods, evaluating DeepSeek-R1-Distill-Qwen-1.5B/7B, and OLMo-3-7B-Think on the PolyMath benchmark across 16 languages and four difficulty levels as representative of a complex, reasoning-heavy, multilingual task. We compare BF16 baselines with several quantized variants, and evaluate both final-answer accuracy and model ability to maintain the correct language (language consistency). We show that quantization affects task accuracy and language consistency differently: (1) AutoRound best preserves accuracy in integer quantization, (2) while GPTQ often preserves language consistency better in reasoning. (3) floating-point quantization robustness is model-dependent. (4) We also coin the "language consistency breakdown effect": harder problems amplify sharp degradation in language consistency, especially for non-Latin or lower-resource languages. We find that evaluating quantized reasoning models by accuracy alone can hide important multilingual effects, and measuring both final-answer correctness and language consistency is critical to understanding the effect of quantization on reasoning models.

1 Introduction

Recent large language models (LLMs) trained with enhanced reasoning abilities have achieved strong performance on mathematical, scientific, and competition-level reasoning tasks (Jaech et al., 2024; Team et al., 2025; Guo et al., 2025; Team, 2025; Rein et al., 2023; Lightman et al., 2024; Cobbe et al., 2021). These gains often rely on long chain-of-thought reasoning (Wei et al., 2022), which improves accuracy but substantially increases inference cost and may cause models to "overthink" simple problems (Chen et al., 2024). Efficient deployment of reasoning LLMs therefore requires compression methods that preserve not only final-answer accuracy, but also the reasoning behavior of the model.

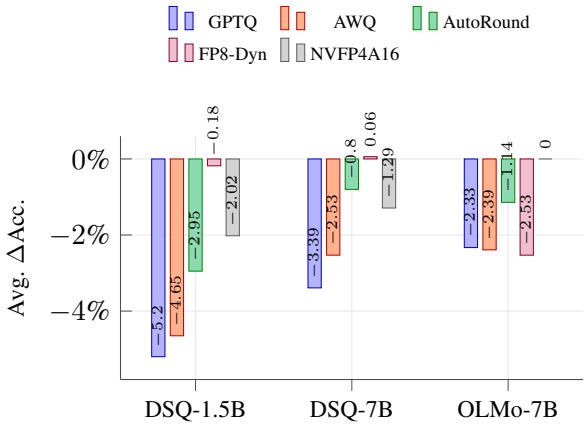

Figure 1: Average accuracy change relative to BF16 across several quantization methods on PolyMath benchmark. Values closer to zero indicate better accuracy preservation. DSQ: DeepSeek-R1-Distill-Qwen; OLMo-7B: OLMo-3-7B-Think

Quantization is one of the most practical approaches for reducing LLM serving and inference cost by lowering the memory usage. It improves the throughput by reducing the precision of weights, activations, or KV cache tensors (Bai et al., 2021; Xiao et al., 2023; Dettmers et al., 2022, 2023; Liu et al., 2024). However, most quantization studies focus on LLMs with English or general-purpose benchmarks, leaving the behavior of quantized reasoning models in multilingual settings underexplored (Li et al., 2025). (Marchisio et al., 2024) are the first to examine the effect of quantization multilingual LLMs across languages and tasks, but

their work was done only on non-reasoning models. We bring the analysis of the effect of quantization on multilinguality to the reasoning era.

Multilingual reasoning introduces additional risks. A model must solve the problem correctly while preserving the requested language in reasoning trace, answer, or both. Quantization error may accumulate across long reasoning steps (Wei et al., 2022), while language-specific factors such as script, tokenizer coverage, writing direction, and resource level may amplify degradation (Ogueji et al., 2022; Team et al., 2022; Khondaker et al., 2023; Pozzobon et al., 2024). Accordingly, evaluating quantized reasoning LLMs for multilingual benchmarks with final-answer accuracy alone may miss important multilingual insights, such as a drift towards answering or reasoning in an unintended language — what we coin "the language consistency breakdown effect."

In this study, we present a empirical study of quantization effects on multilingual reasoning LLMs. We compare BF16 baselines against several quantization schemes (see Figure 1), reporting both task accuracy and language consistency. We analyze across quantization methods, task difficulty, model scaling, and model architecture. We focus on PolyMath (Wang et al., 2025), a multilevel mathematical evaluation, as representative of a challenging task requiring reasoning that is available in many languages.

Our main findings are summarized as follows:

1. **AutoRound performs better amongst W4A16 integer methods for accuracy preservation (§4.1):** AWQ is generally second-best, while GPTQ shows large accuracy drops. This gives the overall accuracy order:

$$\text{AutoRound} > \text{AWQ} \geq \text{GPTQ}.$$

2. **Thinking-language consistency (TC) can be affected by quantization (§4.2, §5.3):** TC can either be improved or degraded after quantization. Using tolerance-based TC votes with $\epsilon = 1.0$, GPTQ often performs better than AWQ and AutoRound.

   This also exposes a *language consistency breakdown effect* (see Figure 2), where models, at harder difficulty levels, shifts the reasoning trace or answer away from the requested language.

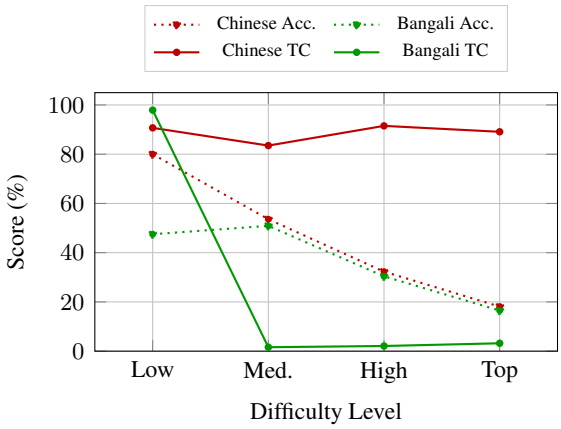

Figure 2: BF16 difficulty-wise accuracy and thinking-language consistency (TC) for Chinese and Bangali on DeepSeek-R1-Distill-Qwen-7B. Dotted lines show accuracy, while solid lines show TC across PolyMath difficulty levels. Chinese maintains strong thinking-language consistency (TC) across difficulty levels, whereas Bangali shows a language consistency breakdown effect in thinking.

3. **Floating-point quantization methods better preserve language consistency, but accuracy remains model-dependent (§4.3, §4.4, §5.5):** FP8-Dynamic is nearly lossless for the DeepSeek Qwen models, unlike for OLMo-3-7B-Think. NVFP4A16 is more reliable than NVFP4. Overall, floating-point formats preserve language consistency better than integer W4A16 methods, but their robustness still depends on quantization precision and the model architecture.

4. **Quantization robustness is not language-neutral (§5.4):** High-resource and Latin-script languages generally preserve accuracy and language consistency more reliably on quantization, whereas lower-resource or non-Latin languages show larger instability and language drift.

Overall, our work highlights that multilingual reasoning models cannot be evaluated by English accuracy and compression alone; multilingual accuracy and language consistency must be treated as important evaluation criteria.

2 Preliminary and Related Work

Quantization Quantization reduces the numerical precision of model weights, activations or KV cache tensors to lower the memory footprints and improve inference throughput. For a tensor $\mathbf{X}$, a

Method	W-A-KV Bits	Weight Format	Activation	Calibration	Core Strategy
Integer (Weight-Only)					
GPTQ	4-16-16	INT4 (g128)	BF16	✓	Hessian Update
AWQ	4-16-16	INT4 (g128)	BF16	✓	Act-aware Scaling
AutoRound	4-16-16	INT4 (g128)	BF16	✓	SignSGD Rounding
Float (Weight + Activation)					
FP8-Dyn	8-8-16	FP8 (E4M3)	FP8, Dynamic	×	Round-to-nearest (RTN)
NVFP4 + AWQ	4-4-16	NVFP4 (g16)	NVFP4, Dynamic	✓	Act-aware Scaling
NVFP4A16 + AWQ	4-16-16	NVFP4 (g16)	BF16	✓	Act-aware Scaling
MXFP4A16 + AWQ	4-16-16	MXFP4 (g32)	BF16	✓	Act-aware Scaling

Table 1: Integer- and float-based quantization configurations. All methods use BF16 KV Cache.

standard b-bit uniform quantization function is defined as:

$$\hat{\mathbf{X}} = \mathcal{Q}(\mathbf{X}; b) = s \cdot \text{clip}\left(\left\lfloor \frac{\mathbf{X}}{s} \right\rceil, q_{\min}, q_{\max}\right), \tag{1}$$

where s is the quantization scale, $\lfloor \cdot \rceil$ denotes the round-to-nearest-integer operation, and $[q_{\min}, q_{\max}]$ represents the b-bit integer clipping range. In this study, we focus on post-training quantization (PTQ), which quantizes a trained model without gradient updates and is widely used for the LLM compression.

Weight-only and Weight-Activation Quantization Methods. In weight-only quantization, activations remain in higher precision while weights are quantized:

$$\hat{\mathbf{Y}} = \mathbf{A}\hat{\mathbf{W}}, \qquad \hat{\mathbf{W}} = \mathcal{Q}(\mathbf{W}; b_w). \tag{2}$$

This reduces memory footprint and bandwidth that lead to improved throughput, but low-bit W4A16 settings require careful methods to preserve model accuracy. We evaluate GPTQ (Frantar et al., 2022), AWQ (Lin et al., 2024), and AutoRound (Cheng et al., 2024). We also evaluate W4A16 floating-point quantized formats, including MXFP4A16 (Rouhani et al., 2023), and NVFP4A16 variants (Abecassis et al., 2025).

Weight-activation quantization also quantizes activations by calculating calibration range and scale values runtime (dynamic) or during calibration stage (static):

$$\hat{\mathbf{Y}} = \hat{\mathbf{A}}\hat{\mathbf{W}}, \quad \hat{\mathbf{A}} = \mathcal{Q}(\mathbf{A}; b_a), \quad \hat{\mathbf{W}} = \mathcal{Q}(\mathbf{W}; b_w). \tag{3}$$

This can also improve throughput but is sensitive to activation outliers (Xiao et al., 2023). We evaluate floating-point quantized formats, including FP8-Dynamic (W8A8) and NVFP4 (W4A4) models.

Reasoning Models Recent reasoning-oriented LLMs, such as OpenAI o1 (Jaech et al., 2024), DeepSeek-R1 (Guo et al., 2025), QwQ (Team, 2025; Yang et al., 2024), and Kimi (Team et al., 2025), have shown great performance on reasoning and multi-step problem solving. These models are typically developed/ improved through supervised fine-tuning, reinforcement, distillation, or hybrid multi-step post-training pipelines (Guo et al., 2025; Muennighoff et al., 2025; Ye et al., 2025; Team, 2025; Team et al., 2025).

Given an input problem x, a reasoning model generates an intermediate reasoning traces r before predicting the final-answer y:

$$p_\theta(y, r \mid x) = p_\theta(r \mid x)\, p_\theta(y \mid x, r). \tag{4}$$

It is observed that reasoning traces can improve the accuracy, they also increase decoding cost, latency, and memory usage (Wei et al., 2022; Chen et al., 2024). Prior work reduces this cost through token budget control, length penalties, and various test-time scaling strategies (Han et al., 2025; Muennighoff et al., 2025; Chen et al., 2024; Team et al., 2025). Quantization is also a complementary method because it reduces model precision without directly reducing the reasoning trace.

Multilingual Reasoning and Quantization Multilingual reasoning requires models to solve problems while preserving the target language in reasoning trace, answer, or both. Prior work

shows that multilingual reasoning LLM performance varies across languages due to training data, tokenizer coverage for target language, and resource level (Ahia et al., 2021; Ogueji et al., 2022; Team et al., 2022; Khondaker et al., 2023; Pozzobon et al., 2024). Recent studies show that quantization can amplify these disparities (Marchisio et al., 2024) and observe strong degradation for non-Latin languages, while (Liu et al., 2025) show that W8A8 and W4A16 can be nearly lossless for reasoning models on English reasoning benchmarks but low bit settings are riskier. Multilingual calibration data can also improve the performance of quantized models and reduce quantization error (Chimoto et al., 2026).

While prior quantization studies largely focus either on multilingual non-reasoning models or English reasoning benchmarks, our work extends these studies on quantized multilingual reasoning using PolyMath (Wang et al., 2025). We evaluate both final-answer accuracy and thinking-language consistency (TC), allowing us to test quantized reasoning models in different settings.

3 Experimental Setup and Dataset

Evaluation Benchmark. We evaluate the baseline reasoning model and corresponding quantized variants on PolyMath benchmark (Wang et al., 2025). It is a multilingual mathematical benchmark to evaluate reasoning models. In this paper, we evaluate 16 languages from PolyMath and study the effect of quantization with task accuracy, each difficulty level accuracies and language consistence scores. This benchmark is selected because it spans across various languages, difficulty levels, and provides standard scripts to calculate final-answer correctness and language consistency scores.

Models. We evaluate three open-source reasoning models including DeepSeek-R1-Distill-Qwen-1.5B, DeepSeek-R1-Distill-Qwen-7B (Guo et al., 2025), and OLMo-3-7B-Think (Olmo et al., 2026). These models are selected to study model scaling effect and to analyze the effect of quantization across two different model architecture.

Inference Setup. All inference is performed using `vLLM` framework. We set the same inference settings used by (Liu et al., 2025). We use a temperature of 0.6 and top-p of 0.95. Our setup differs only in the maximum model sequence length; we use 65,536 (instead of 32,768) because multilingual generations can require more tokens to express the same semantic content. To reduce generation variations, each reasoning model and its quantized variants is evaluated over three random seeds. The prompting strategy is reported in Appendix B.

Quantization Setup. Each quantization method has several tunable choices. It includes calibration data, group size, and weight/activation precision. For a fair and reproducible comparison, we use standardized `llm-compressor` presets as summarized in Table 1. Following (Liu et al., 2025), all inference is performed with `vLLM` using the same decoding configurations and calibration settings. Calibration based methods use 128 samples from the Pile dataset (Gao et al., 2020). We use this general-domain calibration source instead of a multilingual or math-specific set to avoid benchmark-aligned calibration gains, since calibration data can itself affect quantized model performance (Liu et al., 2025; Chimoto et al., 2026).

Evaluation Metrics. A model should preserve the requested language in reasoning trace, answer, or both. Accordingly, in addition to the standard accuracy reported on PolyMath, we evaluate thinking-language consistency (TC) and answer-language consistency (AC). Let $\hat{\ell}_i^r$ denote the detected language of the reasoning trace. We define TC as:

$$\text{TC} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[\hat{\ell}_i^r = \ell_i]. \tag{5}$$

Each quantized model is compared against its BF16 baseline using following Δ values:

$$\begin{aligned} \Delta_{\text{Acc}} &= \text{Acc}_{\text{quant}} - \text{Acc}_{\text{BF16}}, \\ \Delta_{\text{TC}} &= \text{TC}_{\text{quant}} - \text{TC}_{\text{BF16}} \end{aligned} \tag{6}$$

We report average accuracy, difficulty-wise accuracy, and TC scores across languages, model scales, and model families in Table 2 whereas, answer-language consistency (AC) scores are reported in appendix C.2. These metrics allow us to test whether quantized reasoning models preserve both final-answer correctness and target language consistency. Further hardware and software details are reported in appendix D.

4 Results

We report benchmarking accuracies and thinking-language consistency scores in Table 2 and corresponding delta values, relative to BF16 in Table 3.

Model	Method	W-A-KV # Bits	en		zh		ar		bn		fr		ru		sw		te		Avg	
			Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC
DeepSeek-R1 Distill-Qwen-1.5B	—	**BF16**	**35.7**	**99.8**	**30.0**	**89.2**	**18.7**	**28.0**	**15.3**	**7.60**	**24.2**	**45.8**	**21.6**	**66.3**	**7.6**	**20.0**	**9.50**	**25.4**	**20.3**	**47.8**
	GPTQ	4-16-16	29.6	100.0	23.5	88.5	12.3	31.2	9.80	18.2	18.1	56.5	17.0	57.9	6.40	42.7	4.30	58.3	15.1	56.7
	AWQ	4-16-16	29.7	99.8	25.9	85.8	14.5	29.9	9.80	13.9	18.8	48.4	16.7	58.4	4.90	38.0	5.10	39.3	15.7	51.7
	AutoRound	4-16-16	31.1	99.7	23.1	88.5	15.7	31.1	11.3	28.1	21.4	45.6	21.8	45.2	6.70	32.3	7.90	31.7	17.4	50.3
	FP8-Dynamic	8-8-16	34.9	99.6	28.6	87.9	19.3	26.9	15.6	7.60	24.1	47.3	21.9	62.2	7.90	14.7	8.90	27.7	20.2	46.7
	NVFP4	4-4-16	32.9	99.8	27.6	90.3	13.9	45.4	12.2	17.5	18.6	58.1	17.5	77.8	6.5	25.0	8.5	23.3	17.2	54.7
	NVFP4A16	4-16-16	35.7	99.9	28.3	91.3	14.7	46.8	13.5	16.0	20.8	57.2	17.9	76.3	6.2	27.7	9.3	22.9	18.3	54.8
DeepSeek-R1 Distill-Qwen-7B	—	**BF16**	**49.2**	**99.8**	**46.0**	**88.7**	**39.2**	**20.2**	**36.3**	**26.2**	**42.0**	**37.8**	**37.6**	**88.9**	**16.1**	**26.0**	**25.3**	**31.1**	**36.5**	**52.3**
	GPTQ	4-16-16	44.8	99.7	40.3	90.8	33.6	14.5	32.4	19.6	39.8	31.0	37.0	61.4	14.1	27.3	22.6	31.2	33.1	46.9
	AWQ	4-16-16	45.6	99.7	42.0	88.7	36.3	15.5	33.7	21.8	39.5	30.2	34.8	86.6	16.0	20.4	23.6	25.2	33.9	48.5
	AutoRound	4-16-16	48.0	100.0	43.3	87.9	38.8	14.5	32.7	21.3	41.8	26.8	40.2	63.5	15.0	25.8	25.5	24.9	35.7	45.6
	FP8-Dynamic	8-8-16	48.0	99.8	45.8	90.0	40.1	20.8	35.7	25.9	42.8	39.5	38.2	90.7	15.4	26.8	26.2	33.1	36.5	53.3
	NVFP4	4-4-16	38.3	100.0	31.5	89.8	22.1	20.9	16.2	15.5	32.1	30.1	26.1	85.8	9.8	27.3	20.3	26.8	24.6	49.5
	NVFP4A16	4-16-16	47.1	100.0	44.1	89.3	39.0	18.0	34.0	24.0	41.4	31.0	36.5	85.7	14.9	26.5	24.4	27.4	35.2	50.2
Olmo-3-7B-Think	—	**BF16**	**57.6**	**99.7**	**54.0**	**11.3**	**44.7**	**0.5**	**32.9**	**3.1**	**53.9**	**0.4**	**38.1**	**99.4**	**22.9**	**0.7**	**37.6**	**1.1**	**42.7**	**27.0**
	GPTQ	4-16-16	57.4	99.9	49.3	44.2	43.0	1.3	30.1	3.5	52.9	0.6	35.6	98.2	20.4	1.3	34.4	1.4	40.4	31.3
	AWQ	4-16-16	57.4	99.7	51.0	24.2	43.8	1.2	29.2	4.3	51.2	1.6	34.1	98.3	20.7	1.0	35.2	2.2	40.3	29.1
	AutoRound	4-16-16	56.2	99.8	51.5	11.6	46.0	0.6	30.5	2.7	54.2	1.1	37.5	96.9	21.1	0.7	35.6	1.3	41.6	26.8
	FP8-Dynamic	8-8-16	55.9	99.9	50.4	14.8	42.1	3.2	30.1	8.7	50.5	0.5	35.1	99.4	22.0	0.7	35.4	1.0	40.2	28.5
	NVFP4	4-4-16	57.5	100.0	47.2	77.3	38.2	27.7	38.1	1.1	55.6	1.0	37.8	99.4	21.5	0.6	37.9	0.8	41.7	38.5
	NVFP4A16	4-16-16	59.7	100.0	47.8	89.4	38.5	31.6	40.5	1.8	56.0	0.9	40.1	99.7	22.7	0.2	36.4	1.1	42.7	40.6

Table 2: Accuracy and thinking-language consistency (TC) results across eight languages and quantization methods. **Acc.** denotes benchmark accuracy and **TC** denotes thinking-language consistency scores. The green/red cells mark the best/worst accuracy values respectively within each model-language-quantization group and total average column. Other eight languages results are reported in Table 5.

Briefly, quantization methods affects multilingual reasoning performance unevenly: generally quantization methods preserve or worsen final-answer accuracy, but may actually degrade or improve the language consistency scores.

4.1 Accuracy Preservation

Among the weight-only integer quantization methods, AutoRound provides the better accuracy preservation across quantized models. For DeepSeek-R1-Distill-Qwen-1.5B, AutoRound drops the average accuracy by only 2.95 points, compared with 4.65 for AWQ and 5.20 for GPTQ as shown in Table 3. This trend is even better for DeepSeek-R1-Distill-Qwen-7B, where AutoRound reduces the accuracy by 0.80 points, while AWQ and GPTQ drop by 2.53 and 3.39 points respectively as shown in Table 3. Similar trend is observed for the same model with 16 languages results highlighted in Table 5 and Table 6. For OLMo-3-7B-Think, AutoRound again gives the better accuracy degradation (1.14 points), outperforming GPTQ (2.33) and AWQ (2.39) as shown in Table 3. Thus, the overall integer weight-only quantization methods ranking based on our results is:

$$\text{AutoRound} > \text{AWQ} \geq \text{GPTQ}.$$

This indicates that AutoRound is generally a reliable W4A16 integer quantization method for preserving final-answer accuracy in multilingual settings, although the magnitude of its advantage varies across models and languages.

4.2 Thinking-Language Consistency

The quantization order changes when we evaluate thinking-language consistency (TC), since TC can either be improved or degraded after quantization. Avg. ΔTC captures overall TC improvement, but it can be skewed by large gains or drops in a small number of languages. Therefore, we use tolerance-based TC voting as the main criteria to rank quantization methods for TC and report

Model	Method	W-A-KV # Bits	en		zh		ar		bn		fr		ru		sw		te		Avg. Δ	
			ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	Acc.	TC
DeepSeek-R1 Distill-Qwen-1.5B	GPTQ	4-16-16	-6.1	+0.2	-6.5	-0.7	-6.4	+3.2	-5.5	+10.6	-6.1	+10.7	-4.6	-8.4	-1.2	+22.7	-5.2	+32.9	-5.20	+8.90
	AWQ	4-16-16	-6.0	0.0	-4.1	-3.4	-4.2	+1.9	-5.5	+6.3	-5.4	+2.6	-4.9	-7.9	-2.7	+18.0	-4.4	+13.9	-4.65	+3.92
	AutoRound	4-16-16	-4.6	-0.1	-6.9	-0.7	-3.0	+3.1	-4.0	+20.5	-2.8	-0.2	+0.2	-21.1	-0.9	+12.3	-1.6	+6.3	-2.95	+2.51
	FP8-Dynamic	8-8-16	-0.8	-0.2	-1.4	-1.3	+0.6	-1.1	+0.3	0.0	-0.1	+1.5	+0.3	-4.1	+0.3	-5.3	-0.6	+2.3	-0.18	-1.02
	NVFP4	4-4-16	-2.8	+0.0	-2.4	+1.1	-4.8	+17.4	-3.1	+9.9	-5.6	+12.3	-4.1	+11.5	-1.1	+5.0	-1.0	-2.1	-3.11	+6.89
	NVFP4A16	4-16-16	+0.0	+0.1	-1.7	+2.1	-4.0	+18.8	-1.8	+8.4	-3.4	+11.4	-3.7	+10.0	-1.4	+7.7	-0.2	-2.5	-2.02	+7.00
DeepSeek-R1 Distill-Qwen-7B	GPTQ	4-16-16	-4.4	-0.1	-5.7	+2.1	-5.6	-5.7	-3.9	-6.6	-2.2	-6.8	-0.6	-27.5	-2.0	+1.3	-2.7	+0.1	-3.39	-5.40
	AWQ	4-16-16	-3.6	-0.1	-4.0	0.0	-2.9	-4.7	-2.6	-4.4	-2.5	-7.6	-2.8	-2.3	-0.1	-5.6	-1.7	-5.9	-2.53	-3.83
	AutoRound	4-16-16	-1.2	+0.2	-2.7	-0.8	-0.4	-5.7	-3.6	-4.9	-0.2	-11.0	+2.6	-25.4	-1.1	-0.2	+0.2	-6.2	-0.80	-6.75
	FP8-Dynamic	8-8-16	-1.2	0.0	-0.2	+1.3	+0.9	+0.6	-0.6	-0.3	+0.8	+1.7	+0.6	+1.8	-0.7	+0.8	+0.9	+2.0	+0.06	+0.99
	NVFP4	4-4-16	-10.9	+0.2	-14.5	+1.1	-17.1	+0.7	-20.1	-10.7	-9.9	-7.7	-11.5	-3.1	-6.3	+1.3	-5.0	-4.3	-11.91	-2.81
	NVFP4A16	4-16-16	-2.1	+0.2	-1.9	+0.6	-0.2	-2.2	-2.3	-2.2	-0.6	-6.8	-1.1	-3.2	-1.2	+0.5	-0.9	-3.7	-1.29	-2.10
Olmo-3-7B-Think	GPTQ	4-16-16	-0.2	+0.2	-4.7	+32.9	-1.7	+0.8	-2.8	+0.4	-1.0	+0.2	-2.5	-1.2	-2.5	+0.6	-3.2	+0.3	-2.33	+4.28
	AWQ	4-16-16	-0.2	0.0	-3.0	+12.9	-0.9	+0.7	-3.7	+1.2	-2.7	+1.2	-4.0	-1.1	-2.2	+0.3	-2.4	+1.1	-2.39	+2.04
	AutoRound	4-16-16	-1.4	+0.1	-2.5	+0.3	+1.3	+0.1	-2.4	-0.4	+0.3	+0.7	-0.6	-2.5	-1.8	0.0	-2.0	+0.2	-1.14	-0.19
	FP8-Dynamic	8-8-16	-1.7	+0.2	-3.6	+3.5	-2.6	+2.7	-2.8	+5.6	-3.4	+0.1	-3.0	0.0	-0.9	0.0	-2.2	-0.1	-2.53	+1.50
	NVFP4	4-4-16	-0.1	+0.3	-6.8	+66.0	-6.5	+27.2	+5.2	-2.0	+1.7	+0.6	-0.3	+0.0	-1.4	-0.1	+0.3	-0.3	-0.99	+11.46
	NVFP4A16	4-16-16	+2.1	+0.3	-6.2	+78.1	-6.2	+31.1	+7.6	-1.3	+2.1	+0.5	+2.0	+0.3	-0.2	-0.5	-1.2	+0.0	+0.00	+13.56

Table 3: Accuracy and Thinking-language Consistency (TC) Δ scores, relative to BF16 across eight languages. The green/red cells mark the best/worst integer methods by magnitude-based ΔTC scores. Yellow cells indicate tolerance-based TC voting co-winners with $\epsilon = 1.0$, i.e., methods within one TC point of the best ΔTC for each model-language-quantization group are considered as co-winner. Table 4 reports tolerance-based voting results for $\epsilon = 1.0$.

avg. ΔTC as a complementary magnitude measure. The tolerance parameter ϵ defines the minimum TC difference considered meaningful, preventing very small gaps from being over-interpreted. For each model-language pair, an integer W4A16 quantization method $m \in \mathcal{M}$ is counted as a co-winner if

$$\Delta TC_m \geq \max_{m' \in \mathcal{M}} \Delta TC_{m'} - \epsilon,$$

where $\mathcal{M}$ denotes GPTQ, AWQ, and AutoRound, and we set $\epsilon = 1.0$ TC point as the meaningful-difference threshold.

Under this tolerance-based criterion, GPTQ receives the better overall support for TC improvement than AWQ and AutoRound as shown in Table 4. The avg. ΔTC values further show that GPTQ has the better TC gains for DeepSeek-R1-Distill-Qwen-1.5B and OLMo-3-7B-Think as shown in Table 3. Appendix C.1 reports further tolerance-based voting results for $\epsilon \in \{0.5, 1.0, 2.0\}$.

4.3 FP8 vs. BF16.

FP8-Dynamic is a calibration-free quantization method. For DeepSeek-R1-Distill-Qwen-1.5B, FP8 drops small average accuracy of 0.18 points. For DeepSeek-R1-Distill-Qwen-7B, FP8 slightly improves the average accuracy by 0.06 points and TC by 0.99 points highlighted in Table 3. However, OLMo-3-7B-Think shows a large FP8 accuracy drop of 2.53 points, lower than all the quantization methods for this model. These results indicate that **FP8 performance is model-dependent**.

4.4 NVFP4 vs. NVFP4A16

As highlighted in Tables 2 and 3, NVFP4A16 (W4A16) generally provides a better accuracy–TC trade-off than NVFP4 (W4A4).

This behavior is likely due to activation sensitivity, full NVFP4 quantizes both weights and activations, making the model more vulnerable to activation outliers. However, NVFP4A16 avoids this error by keeping activations in higher precision. Another interesting insight is that the floating-point methods used in this paper preserve thinking-language consistency more reliably than integer methods, although their accuracy robustness remains model-dependent.

5 Analyses and Discussion

In this section, we analyze how quantization interacts in different multilingual settings, including task difficulty, language consistency, and model

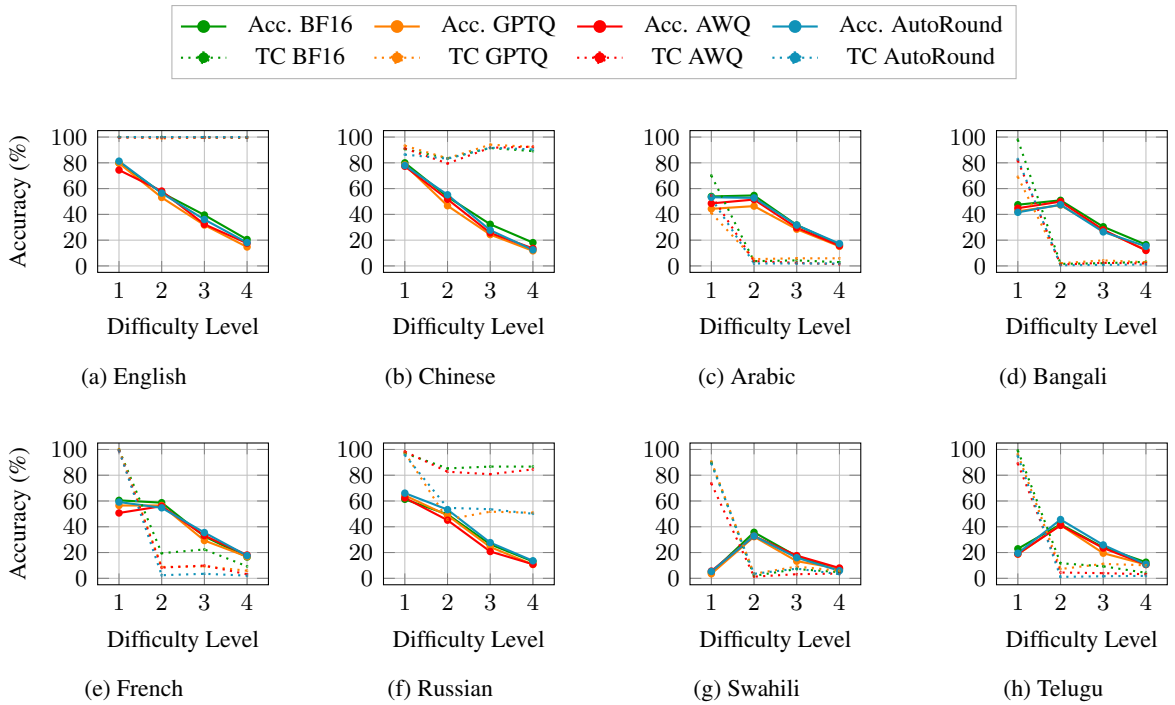

Figure 3: Difficulty-wise accuracy and thinking-language consistency (TC) of DeepSeek-R1-Distill-Qwen-7B across eight languages. Solid lines show accuracy while dotted lines show TC across four difficulty levels. The results show that quantization effects vary by language and difficulty levels: 1 = low, 2 = medium, 3 = high, and 4 = top. Accuracy generally decreases as problem difficulty increases, while TC fluctuates sharply, especially in lower-resource languages, revealing a *language consistency breakdown effect*.

scale. While the Tables 2 and 3 show average accuracy and thinking-language consistency scores across languages; the task difficulty, language consistency breakdown and model scaling reveal more nuanced picture.

5.1 Impact of Model Size

How does model scaling influence the performance of quantized reasoning models?

Model scaling improves multilingual reasoning accuracy and also increases the robustness to quantization. As shown in Table 2, DeepSeek-R1-Distill-Qwen-7B outperforms the corresponding 1.5B model across most languages under both BF16 and quantized variants. For example, AutoRound causes only a 0.80 point average accuracy drop for the 7B model, compared with 2.95 points for 1.5B model in Table 3. Similar trend is observed for GPTQ and AWQ for these two models.

These results suggest that **small reasoning models are more sensitive to low-bit compression**, and large models show robustness toward quantization degradation.

5.2 Impact of Quantization on Task Difficulty

How does task difficulty affect quantized reasoning models?

It is observed that the task difficulty generally increases quantization degradation. As shown in Figures 7 and 8, the relative performance degradation (RPD) grows on harder problems, especially for GPTQ and AWQ. For DeepSeek-R1-Distill-Qwen-7B, AutoRound remains close to BF16 on `low` and `medium` difficulty levels and degrades mainly on the `top` difficulty level. However, GPTQ shows a consistent increase in degradation as difficulty rises. For OLMo-3-7B-Think, GPTQ and AWQ show their large degradation on `high` and `top` difficulty levels, whereas AutoRound is comparatively more stable.

This trend is not universal. AWQ and NVFP4A16 on DeepSeek-R1-Distill-Qwen-7B, and NVFP4A16 on OLMo-3-7B-Think, show irregular patterns in task difficulty. These exceptions suggest that quantization sensitivity pattern for task difficulty also depends on model architecture, quantization method, numerical format, and other factors. Overall, **harder problems are generally more sensitive to quantization**, likely

because they require longer reasoning traces and accumulate more quantization error.

5.3 Language Consistency Breakdown Effect

How does language consistency breakdown effect impact multilingual reasoning model?

In this study, we also reveal a *language consistency breakdown effect* in the reasoning trace for DeepSeek-R1-Distill-Qwen-1.5B and 7B, shown in Figures 3, 4, and 6. In several languages, especially Arabic, Bangali, Indonesian, Japanese and Telugu, TC scores drop sharply as difficulty level increases. This suggests that the model can switch to a more dominant internal reasoning language automatically on harder problems, against the requested thinking language. As a result, higher accuracy at harder difficulty levels can sometimes appear due to a language consistency breakdown in the reasoning trace. This effect becomes visible when accuracy, thinking-language consistency, and difficulty levels are analyzed together. Swahili in Figure 5, being very low-resource language, shows exception that does not follow task-difficulty effect or language consistency breakdown effect. This effect is also observed in the answer-language consistency (AC) scores, reported in the appendix C.2.

5.4 Language-Specific Quantization Robustness

Does quantization affect languages differently by script and resource level?

Quantization robustness varies substantially across all languages. As shown in Tables 2 and 5, English performs stable, maintaining near-perfect accuracy and TC across BF16 and quantized variants. Several high-resource or Latin-script languages, such as French, German, Spanish, and Portuguese, also retain strong accuracy than many low-resource languages. However, their TC is still dependent on quantization method and model.

Lower-resource or non-Latin languages show mixed behavior. Arabic, Bangali, Swahili, Telugu, Korean, Thai, and Vietnamese often exhibit lower or more variable TC, even with better average accuracy. At the same time, script alone does not completely explain the results. Chinese and Russian maintain relatively strong TC in several multilingual settings, whereas some Latin-script languages still show TC instability.

5.5 MXFP4A16 versus NVFP4A16

How do FP4 style quantization affect multilingual reasoning?

We compare MXFP4A16 and NVFP4A16 to study the behavior of different FP4 style quantization methods. As shown in Table 7 and 8, NVFP4A16 generally preserves accuracy better than MXFP4A16. For OLMo-3-7B-Think, NVFP4A16 matches the BF16 average accuracy for eight languages, while MXFP4A16 drops by 1.08 points. For DeepSeek-R1-Distill-Qwen-7B, the gap gets larger with NVFP4A16 drops by 1.29 points, whereas MXFP4A16 drops by 9.31 points.

The TC trend is more interesting. For OLMo-3-7B-Think, both NVFP4A16 and MXFP4A16 significantly improve average TC. For DeepSeek-R1-Distill-Qwen-7B, MXFP4A16 and NVFP4A16 reduces TC scores slightly, but this comes with much larger degradation in accuracy. Overall, NVFP4A16 provides the better accuracy and TC trade-off, while both FP4 style formats nearly preserve or improve thinking-language consistency.

6 Conclusion & Future Work

In this paper, we study how quantization affects multilingual reasoning models, focusing on both final-answer accuracy and language consistency in reasoning and answer. Using PolyMath across 16 languages, three reasoning models, two model families, three random seeds, and seven quantization presets, we find that quantization effects are uneven across models, languages, and metrics. In integer quantization methods, AutoRound best preserves accuracy in W4A16 integer-based quantization methods, while GPTQ is often better in maintaining thinking-language consistency. FP8 shows model-dependent behavior. While large reasoning models are more robust to quantization.

Our results also show that accuracy alone is insufficient for evaluating quantized multilingual reasoning. It hides important multilingual effects such as language consistency breakdown. Quantized multilingual reasoning models should be evaluated through both mathematical correctness and language consistence.

Future work should extend this study to more model architectures, large reasoning models, additional quantization methods, other low-bit formats such as MXFP4, and more under-represented languages.

7 Limitations

Model coverage. We evaluate three open reasoning models, DeepSeek-R1-Distill-Qwen-1.5B, DeepSeek-R1-Distill-Qwen-7B, and OLMo-3-7B-Think. The findings may not be generalized to larger models or closed-source systems.

Language coverage. We use 16 languages from PolyMath excluding languages. Quantization effects may be different for other languages.

Prompt dependence. Our experiments use native-language for reasoning and boxed-answer instructions. Different prompting styles such as English reasoning prompts for all languages, free-form reasoning, or few-shot may produce different behavior.

Quantization scope. Our analysis focuses mainly on BF16, GPTQ, AWQ, AutoRound, FP8, NVFP4, NVFP4A16 and MXFP4A16. Further work is needed for other quantization methods such as MXFP4, NVFP4, KV cache quantization.

References

Felix Abecassis, Anjulie Agrusa, Dong Ahn, Jonah Alben, Stefania Alborghetti, Michael Andersch, Sivakumar Arayandi, Alexis Bjorlin, Aaron Blakeman, Evan Briones, and 1 others. 2025. Pretraining large language models with nvfp4. arXiv preprint arXiv:2509.25149.

Orevaoghene Ahia, Julia Kreutzer, and Sara Hooker. 2021. The low-resource double bind: An empirical study of pruning for low-resource machine translation. Preprint, arXiv:2110.03036.

Haoli Bai, Wei Zhang, Lu Hou, Lifeng Shang, Jin Jin, Xin Jiang, Qun Liu, Michael Lyu, and Irwin King. 2021. Binarybert: Pushing the limit of bert quantization. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 4334–4348.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, and 1 others. 2024. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. arXiv preprint arXiv:2412.21187.

Wenhua Cheng, Weiwei Zhang, Haihao Shen, Yiyang Cai, Xin He, Lv Kaokao, and Yi Liu. 2024. Optimize weight rounding via signed gradient descent for the quantization of llms. In Findings of the Association for Computational Linguistics: EMNLP 2024, pages 11332–11350.

Everlyn Asiko Chimoto, Mostafa Elhoushi, and Bruce A Bassett. 2026. Calibrating beyond english: Language diversity for better quantized multilingual llm. arXiv preprint arXiv:2601.18306.

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

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. Advances in neural information processing systems, 35:30318–30332.

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

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

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, and 1 others. 2020. The pile: An 800gb dataset of diverse text for language modeling. arXiv preprint arXiv:2101.00027.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv preprint arXiv:2501.12948.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2025. Token-budget-aware llm reasoning. In Findings of the Association for Computational Linguistics: ACL 2025, pages 24842–24855.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. arXiv preprint arXiv:2412.16720.

Md Tawkat Islam Khondaker, Abdul Waheed, Muhammad Abdul-Mageed, and 1 others. 2023. Gptaraeval: A comprehensive evaluation of chatgpt on arabic nlp. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 220–247.

Zhen Li, Yupeng Su, Runming Yang, Congkai Xie, Zheng Wang, Zhongwei Xie, Ngai Wong, and Hongxia Yang. 2025. Quantization meets reasoning: Exploring llm low-bit quantization degradation for mathematical reasoning. arXiv preprint arXiv:2501.03035.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In International Conference on Learning Representations, volume 2024, pages 39578–39601.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. Proceedings of machine learning and systems, 6:87–100.

Ruikang Liu, Haoli Bai, Haokun Lin, Yuening Li, Han Gao, Zhengzhuo Xu, Lu Hou, Jun Yao, and Chun Yuan. 2024. Intactkv: Improving large language model quantization by keeping pivot tokens intact. arXiv preprint arXiv:2403.01241.

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

Kelly Marchisio, Saurabh Dash, Hongyu Chen, Dennis Aumiller, Ahmet Üstün, Sara Hooker, and Sebastian Ruder. 2024. How does quantization affect multilingual llms? Preprint, arXiv:2407.03211.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. 2025. s1: Simple test-time scaling. In Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 20286–20332.

Kelechi Ogueji, Orevaoghene Ahia, Gbemileke Onilude, Sebastian Gehrmann, Sara Hooker, and Julia Kreutzer. 2022. Intriguing properties of compression on multilingual models. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 9092–9110, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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, and 50 others. 2026. Olmo 3. Preprint, arXiv:2512.13961.

Luiza Pozzobon, Patrick Lewis, Sara Hooker, and Beyza Ermis. 2024. From one to many: Expanding the scope of toxicity mitigation in language models. Preprint, arXiv:2403.03893.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. arXiv preprint arXiv:2311.12022.

Bita Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, Stosic Dusan, Venmugil Elango, Maximilian Golub, Alexander Heinecke, Phil James-Roxby, Dharmesh Jani, Gaurav Kolhe, Martin Langhammer, Ada Li, and 14 others. 2023. Microscaling data formats for deep learning. Preprint, arXiv:2310.10537.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Feng Tang, Flood Sung, Guangda Wei, Guokun Lai, and 75 others. 2025. Kimi k1.5: Scaling reinforcement learning with llms. ArXiv, abs/2501.12599.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, and 20 others. 2022. No language left behind: Scaling human-centered machine translation. Preprint, arXiv:2207.04672.

Qwen Team. 2025. Qwq-32b: Embracing the power of reinforcement learning.

vLLM Team. 2025. Fp8 w8a8 quantization. https://docs.vllm.ai/en/stable/features/quantization/fp8/. VLLM documentation.

Yiming Wang, Pei Zhang, Jialong Tang, Haoran Wei, Baosong Yang, Rui Wang, Chenshu Sun, Feitong Sun, Jiran Zhang, Junxuan Wu, Qiqian Cang, Yichang Zhang, Fei Huang, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Polymath: Evaluating mathematical reasoning in multilingual contexts. arXiv preprint arXiv:2504.18428.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems, 35:24824–24837.

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

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2024. Qwen2.5 technical report. arXiv preprint arXiv:2412.15115.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. Limo: Less is more for reasoning. Preprint, arXiv:2502.03387.

A Quantization Algorithms and Implementation Details

In this section, we briefly describe the quantization methods, evaluated in this study. All quantized models are produced using `llm-compressor` standard presets to ensure a fair and reproducible comparison across methods.

A.1 Integer W4A16 Quantization

GPTQ. GPTQ (Frantar et al., 2022) is a post-training quantization method that reduces reconstruction error between full precision and quantized layer outputs of the model. Given input activation $\mathbf{X}$ and weight $\mathbf{W}$, GPTQ tries to calculate quantized weight matrix $\hat{\mathbf{W}}$ by solving:

$$\arg\min_{\hat{\mathbf{W}}} \|\hat{\mathbf{W}}\mathbf{X} - \mathbf{W}\mathbf{X}\|_F.$$

It uses approximate second-order information to reduce quantization error. In this study, GPTQ is evaluated using the `llm-compressor` W4A16 preset with group size g=128.

AWQ. AWQ (Lin et al., 2024) is an activation-aware post-training quantization method that protects salient weight channels by applying channel-wise scale factor before applying quantization. Given activations $\mathbf{X}$, weights $\mathbf{W}$, and scaling vector $\mathbf{s}$, AWQ rewrites the linear operation as:

$$\mathbf{Y} = (\mathbf{X}\mathbf{s}^{-1})(\mathbf{s}\mathbf{W}^{\top}),$$

where $\mathbf{s}$ is chosen to reduce quantization error due to activation outliers. We evaluate AWQ using `llm-compressor` W4A16 preset with group size g=128. Like GPTQ, we use calibration with 128 samples from the Pile (Gao et al., 2020).

AutoRound. AutoRound (Cheng et al., 2024) is a post-training quantization method that uses signed gradient descent to optimize the weight rounding and clipping ranges values. Instead of relying on nearest rounding, it tries to learn the rounding decisions that reduce layer-wise reconstruction error with no extra inference time overhead. In our paper, we evaluate AutoRound using `llm-compressor` W4A16 preset with group size g=128.

A.2 Floating-Point Quantization

FP8-Dynamic. FP8-Dynamic (vLLM Team, 2025) quantizes each weights to FP8 precision using RTN (Round-to-nearest) method and applies dynamic per-tensor scaling to activations during inference stage. In our experiment, we use the `llm-compressor` FP8 dynamic quantization (W8A8). It does not require calibration data.

NVFP4 NVFP4 (Abecassis et al., 2025) is a 4-bit floating point format designed for fast inference with fine-grained microscaling. It represents values using an FP4-style E2M1 format and applies shared scaling over 16 value micro blocks. In our setup, NVFP4 quantizes both weights and activations, while NVFP4A16 keeps activations in BF16. Both settings use `llm-compressor` presets with microscale group size g=16. This comparison helps to study the importance of activation precision under FP4-style quantization.

MXFP4. MXFP4 (Rouhani et al., 2023) is microscaling format that also uses shared scaling factors over low-precision small groups to capture local dynamic range. In this work, we evaluate MXFP4A16 using `llm-compressor` preset (MXFP4A16), while activations and KV cache remain in BF16.

B Prompting Strategy.

For each problem, the model is instructed to reason in the target language and the target language is again used to place the final-answer inside a boxed expression. These thinking and answer formatting phrases for given target language, are also provided by PolyMath (Wang et al., 2025). The prompting design is given below

```
{question}
{target_language_reasoning}
{target_language_answer_formatting}
```

For English, the prompt becomes:

```
{question}
Use English to think and answer.
Note: Please put the final answer in
\boxed{}.
```

During post-processing, we split the model response into corresponding thinking and answer parts using the `</think>` delimiter if it is available. This way, we record raw response and corresponding reasoning and answer parts for the evaluations.

Model	ϵ	Method	Co-winner Ranking	Mean ΔTC
DeepSeek-R1-Distill-Qwen-1.5B	0.5	GPTQ	7	8.90
		AWQ	2	3.92
		AutoRound	4	2.51
DeepSeek-R1-Distill-Qwen-7B	0.5	GPTQ	10	-6.96
		AWQ	5	-5.99
		AutoRound	5	-5.43
OLMo-3-7B-Think	0.5	GPTQ	5	4.28
		AWQ	7	2.04
		AutoRound	1	-0.19
DeepSeek-R1-Distill-Qwen-1.5B	1.0	GPTQ	7	8.90
		AWQ	2	3.92
		AutoRound	4	2.51
DeepSeek-R1-Distill-Qwen-7B	1.0	GPTQ	11	-6.96
		AWQ	6	-5.99
		AutoRound	6	-5.43
OLMo-3-7B-Think	1.0	GPTQ	7	4.28
		AWQ	7	2.04
		AutoRound	5	-0.19
DeepSeek-R1-Distill-Qwen-1.5B	2.0	GPTQ	7	8.90
		AWQ	3	3.92
		AutoRound	4	2.51
DeepSeek-R1-Distill-Qwen-7B	2.0	GPTQ	11	-6.96
		AWQ	6	-5.99
		AutoRound	8	-5.43
OLMo-3-7B-Think	2.0	GPTQ	8	4.28
		AWQ	7	2.04
		AutoRound	7	-0.19

Table 4: Sensitivity of tolerance-based TC voting to ϵ. Methods within ϵ TC points of the best signed ΔTC are considered as co-winners. DeepSeek-R1-Distill-Qwen-1.5B and OLMo-3-7B-Think values are recorded for eight languages whereas sixteen languages are used in the evaluation for DeepSeek-R1-Distill-Qwen-7B highlighted in Tables 3 and 6.

C Additional Experiments

C.1 Tolerance-Based Voting

We repeat tolerance-based voting with $\epsilon \in \{0.5, 1.0, 2.0\}$. For each model-language pair, the method with the highest signed ΔTC is selected as the reference, and all methods within ϵ TC points are counted as co-winners. We report co-winner counts and average signed ΔTC in Table 4.

C.2 Answer-Language Consistency

In addition to thinking-language consistency (TC), we also report answer-language consistency (AC) to evaluate whether the model preserves the requested language in answer or not. Let $\hat{\ell}_i^a$ denote the detected language of the answer for example i. We define AC as:

$$\mathrm{AC} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[\hat{\ell}_i^a = \ell_i]. \quad (7)$$

As shown in Figures 9 and 10, AC can drop sharply as the difficulty level increases. This indicates that as the problem becomes more difficult, the model may still solve it while shifting the answer language away from the requested language.

D Hardware and Software Details

Our experimental pipeline consists of three stages. quantization, inference, and evaluation.

For quantization, we use `llm-compressor` version 0.9.0.2 and `compressed-tensors` version 0.13.0. For inference, we use `vLLM` version 0.20.1 with `torch` version 2.11.0 and `transformers` version 5.7.0. Evaluation is performed using the official PolyMath scripts, which compute final-answer correctness as well as language consistency scores.

All experiments are run on NVIDIA H100 GPUs, except for NVFP4, NVFP4A16, and MXFP4A16 experiments, which are run on NVIDIA B200 GPUs due to hardware support for FP4-style formats.

E AI Assistance

ChatGPT-5.5 was used for grammar refinement and latex coding assistance with table/figure structure and debugging. The authors take full responsibility for the final published version.

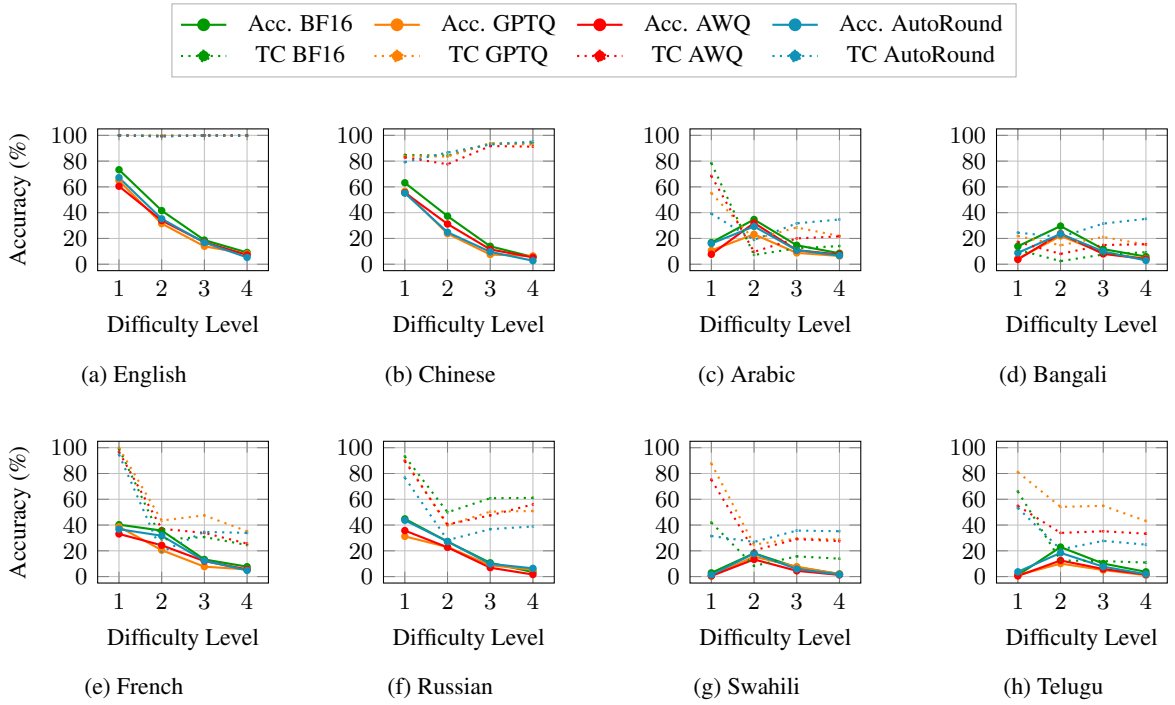

Figure 4: Difficulty-wise accuracy and thinking-language consistency (TC) of DeepSeek-R1-Distill-Qwen-1.5B across eight languages. Solid lines show accuracy while dotted lines show TC across four difficulty levels: 1 = low, 2 = medium, 3 = high, and 4 = top.

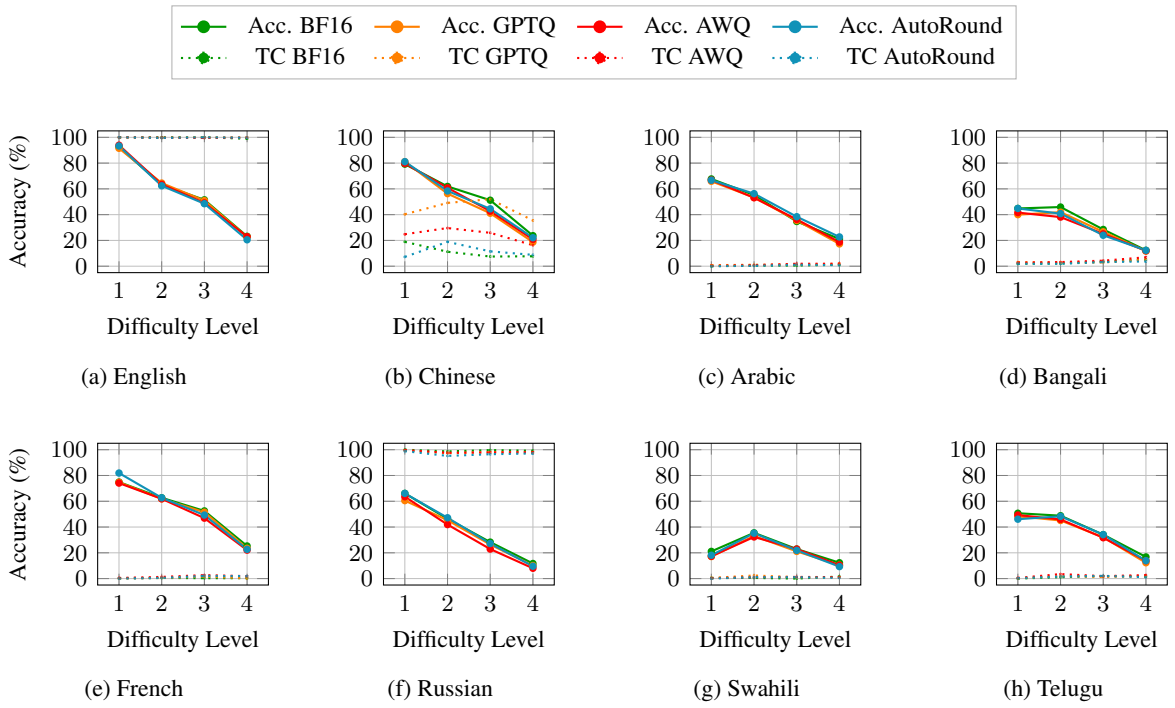

Figure 5: Difficulty-wise accuracy and thinking-language consistency (TC) of Olmo-3-7B-Think across eight languages. Solid lines show accuracy while dotted lines show TC across four difficulty levels: 1 = low, 2 = medium, 3 = high, and 4 = top.

Model	Method	W-A-KV # Bits	de		es		id		ja		ko		pt		th		vi		Avg.	
			Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC
DeepSeek-R1 Distill-Qwen-7B	—	**BF16**	**34.2**	**75.9**	**42.4**	**32.5**	**39.6**	**26.6**	**34.8**	**32.1**	**38.2**	**13.5**	**42.0**	**31.2**	**34.2**	**23.0**	**31.4**	**88.8**	**36.78**	**46.39**
	GPTQ	4-16-16	35.3	41.2	40.3	33.2	37.5	27.1	28.2	43.3	35.1	7.6	40.4	33.6	33.6	20.5	33.7	48.9	34.3	39.4
	AWQ	4-16-16	33.4	59.4	39.8	26.1	36.5	26.9	34.8	26.2	36.0	7.7	40.5	28.1	34.5	15.2	31.9	68.8	34.9	40.4
	AutoRound	4-16-16	36.0	67.1	42.0	28.9	38.6	24.9	34.4	31.7	37.9	11.0	41.3	27.2	36.6	19.1	33.1	80.9	36.6	40.9
	FP8-Dynamic	8-8-16	34.7	78.2	42.4	33.0	39.2	27.4	33.5	33.4	38.9	13.1	42.5	30.7	34.2	24.4	33.1	91.3	36.92	47.38

Table 5: The average accuracy across 8 other languages for DeepSeek-R1-Distill-Qwen-7B. **Acc.** denotes Accuracy and **TC** denotes Thinking-language Consistency. The rightmost **Avg.** column denotes the arithmetic mean across all 16 evaluated languages by combining Table 2 accuracies and TC values. The green/red cells mark the best/worst accuracy values respectively within each model-language-quantization group and total average column.

Model	Method	W-A-KV # Bits	de		es		id		ja		ko		pt		th		vi		Avg.	
			ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC
DeepSeek-R1 Distill-Qwen-7B	GPTQ	4-16-16	+1.1	-34.7	-2.1	+0.7	-2.1	+0.5	-6.6	+11.2	-3.1	-5.9	-1.6	+2.4	-0.6	-2.5	+2.3	-39.9	-2.49	-6.96
	AWQ	4-16-16	-0.8	-16.5	-2.6	-6.4	-3.1	+0.3	0.0	-5.9	-2.2	-5.8	-1.5	-3.1	+0.3	-7.8	+0.5	-20.0	-1.85	-5.99
	AutoRound	4-16-16	+1.8	-8.8	-0.4	-3.6	-1.0	-1.7	-0.4	-0.4	-0.3	-2.5	-0.7	-4.0	+2.4	-3.9	+1.7	-7.9	-0.21	-5.43
	FP8-Dynamic	8-8-16	+0.5	+2.3	0.0	+0.5	-0.4	+0.8	-1.3	+1.3	+0.7	-0.4	+0.5	-0.5	0.0	+1.4	+1.7	+2.5	+0.14	+0.99

Table 6: Delta values for DeepSeek-R1-Distill-Qwen-7B. The rightmost **Avg.** column represents the global mean delta calculated across all 16 evaluated languages by combining Table 3 delta accuracies and TC values. The green/red cells mark the best/worst integer methods by magnitude-based ΔTC scores. Yellow cells indicate tolerance-based TC voting co-winners with $\epsilon = 1.0$, i.e., methods within one TC point of the best ΔTC for each model-language-quantization group are considered as co-winner.

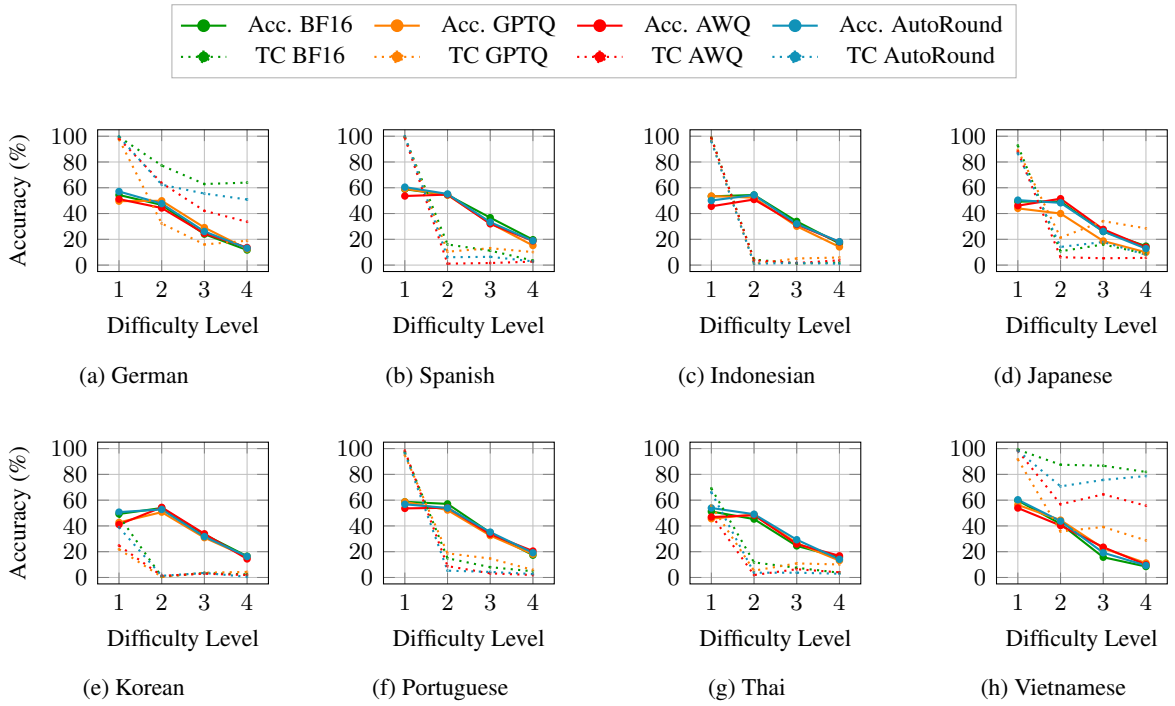

Figure 6: Difficulty-wise accuracy and thinking-language consistency (TC) of DeepSeek-R1-Distill-Qwen-7B other eight languages. Solid lines show accuracy while dotted lines show TC across four difficulty levels: 1 = low, 2 = medium, 3 = high, and 4 = top.

Method	W-A-KV # Bits	en		zh		ar		bn		fr		ru		sw		te		Avg.	
		Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC	Acc.	TC
Olmo-3-7B-Think																			
—	**BF16**	**57.6**	**99.7**	**54.0**	**11.3**	**44.7**	**0.5**	**32.9**	**3.1**	**53.9**	**0.4**	**38.1**	**99.4**	**22.9**	**0.7**	**37.6**	**1.1**	**42.71**	**27.03**
NVFP4A16	4-16-16	59.7	100.0	47.8	89.4	38.5	31.6	40.5	1.8	56.0	0.9	40.1	99.7	22.7	0.2	36.4	1.1	42.71	40.59
MXFP4A16	4-16-16	58.0	99.9	46.5	90.0	37.3	32.0	37.9	0.5	54.6	0.6	39.7	99.5	23.2	0.9	35.9	1.1	41.64	40.56
DeepSeek-R1-Distill-Qwen-7B																			
—	**BF16**	**49.2**	**99.8**	**46.0**	**88.7**	**39.2**	**20.2**	**36.3**	**26.2**	**42.0**	**37.8**	**37.6**	**88.9**	**16.1**	**26.0**	**25.3**	**31.1**	**36.46**	**52.34**
NVFP4A16	4-16-16	47.1	100.0	44.1	89.3	39.0	18.0	34.0	24.0	41.4	31.0	36.5	85.7	14.9	26.5	24.4	27.4	35.17	50.24
MXFP4A16	4-16-16	41.8	100.0	37.7	88.1	26.2	24.8	22.2	22.9	33.6	38.1	28.6	96.0	12.1	17.0	15.0	22.5	27.15	51.17

Table 7: Comparison of NVFP4A16 and MXFP4A16 quantization for Olmo-3-7B-Think and DeepSeek-R1-Distill-Qwen-7B across eight languages. **Acc.** denotes Accuracy and **TC** denotes Thinking-language Consistency. The green/red cells mark the best/worst accuracy values respectively within each model-language-quantization group and total average column.

Method	W-A-KV # Bits	en		zh		ar		bn		fr		ru		sw		te		Avg. Δ	
		ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	ΔAcc.	ΔTC	Acc.	TC
Olmo-3-7B-Think																			
NVFP4A16	4-16-16	+2.1	+0.3	-6.2	+78.1	-6.2	+31.1	+7.6	-1.3	+2.1	+0.5	+2.0	+0.3	-0.2	-0.5	-1.2	+0.0	+0.00	+13.56
MXFP4A16	4-16-16	+0.4	+0.2	-7.5	+78.7	-7.4	+31.5	+5.0	-2.6	+0.7	+0.2	+1.6	+0.1	+0.3	+0.2	-1.7	+0.0	-1.08	+13.54
DeepSeek-R1-Distill-Qwen-7B																			
NVFP4A16	4-16-16	-2.1	+0.2	-1.9	+0.6	-0.2	-2.2	-2.3	-2.2	-0.6	-6.8	-1.1	-3.2	-1.2	+0.5	-0.9	-3.7	-1.29	-2.10
MXFP4A16	4-16-16	-7.4	+0.2	-8.3	-0.6	-13.0	+4.6	-14.1	-3.3	-8.4	+0.3	-9.0	+7.1	-4.0	-9.0	-10.3	-8.6	-9.31	-1.16

Table 8: Delta comparison of NVFP4A16 and MXFP4A16 relative to BF16 for Olmo-3-7B-Think and DeepSeek-R1-Distill-Qwen-7B. The green cells mark the best integer methods by magnitude-based ΔTC scores. Yellow cells indicate tolerance-based TC voting co-winners with $\epsilon = 1.0$, i.e., methods within one TC point of the best ΔTC for each model-language-quantization group are considered as co-winner

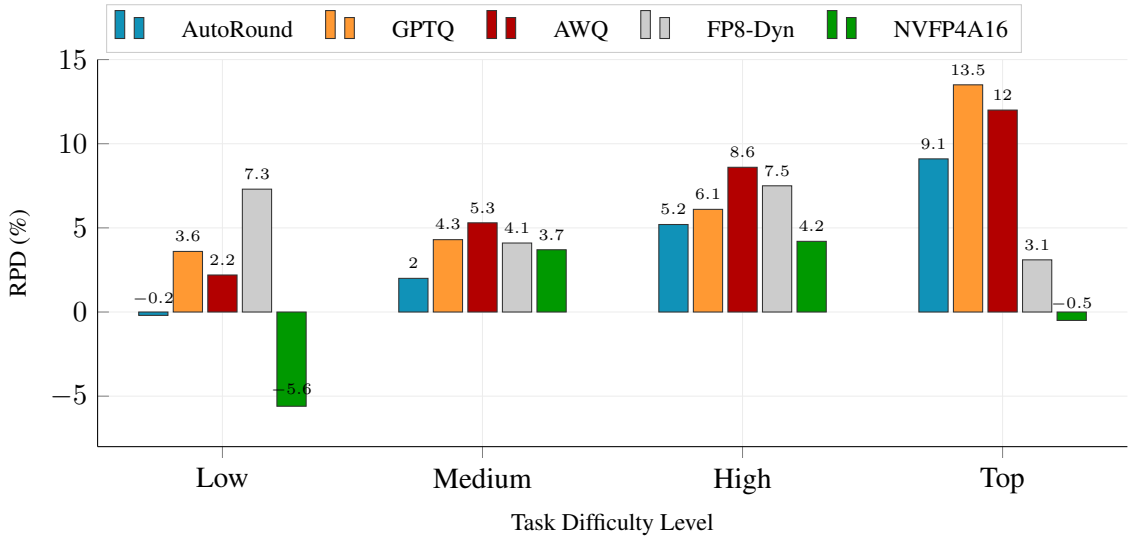

Figure 7: Difficulty-wise Relative Performance Degradation (RPD %) for Olmo-3-7B-Think across eight languages. Lower values indicate better accuracy preservation, with negative values showing gains over BF16.

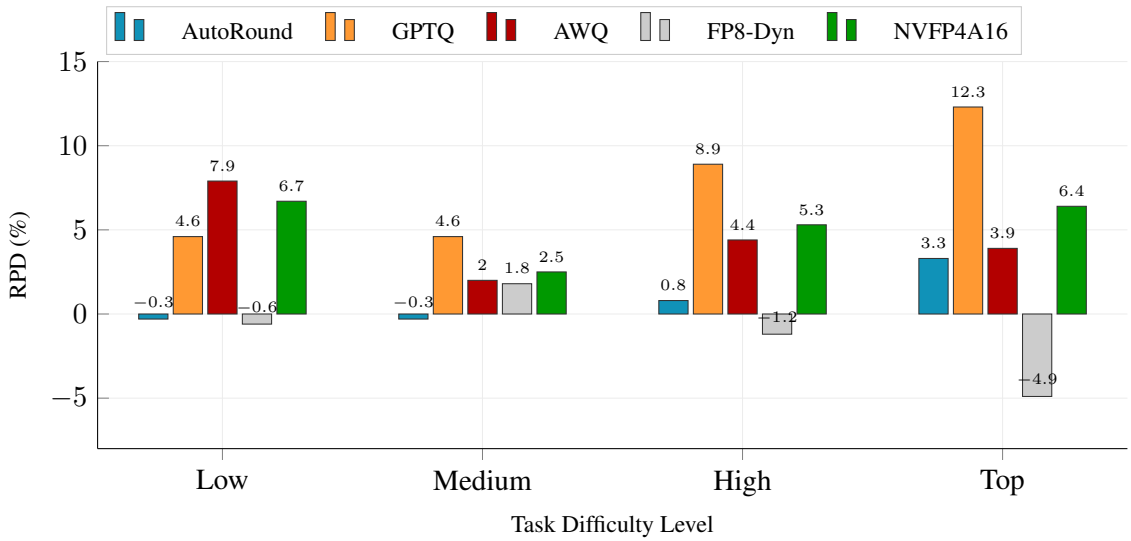

Figure 8: Difficulty-wise Relative Performance Degradation (RPD %) for DeepSeek-R1-Distill-Qwen-7B across 16 languages. Lower values indicate better accuracy preservation, with negative values showing gains over BF16.

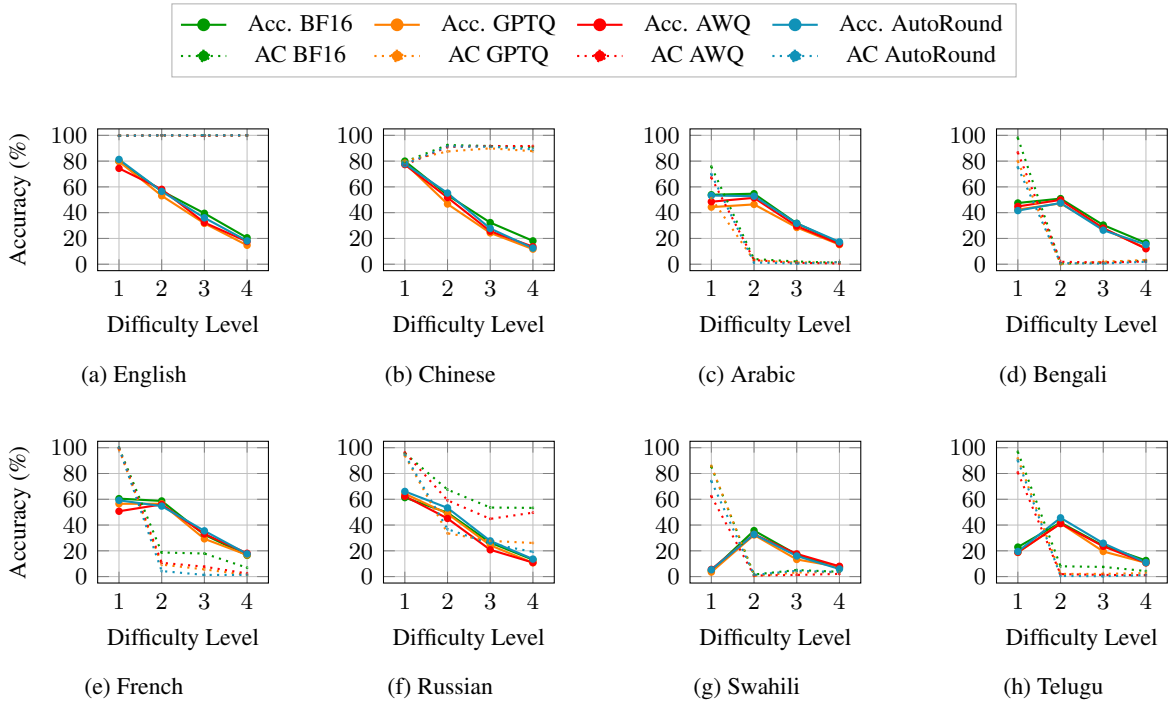

Figure 9: Difficulty-wise accuracy and answer-language consistency (AC) of DeepSeek-R1-Distill-Qwen-7B across eight languages. Solid lines show accuracy while dotted lines show AC across four difficulty levels: 1 = low, 2 = medium, 3 = high, and 4 = top.

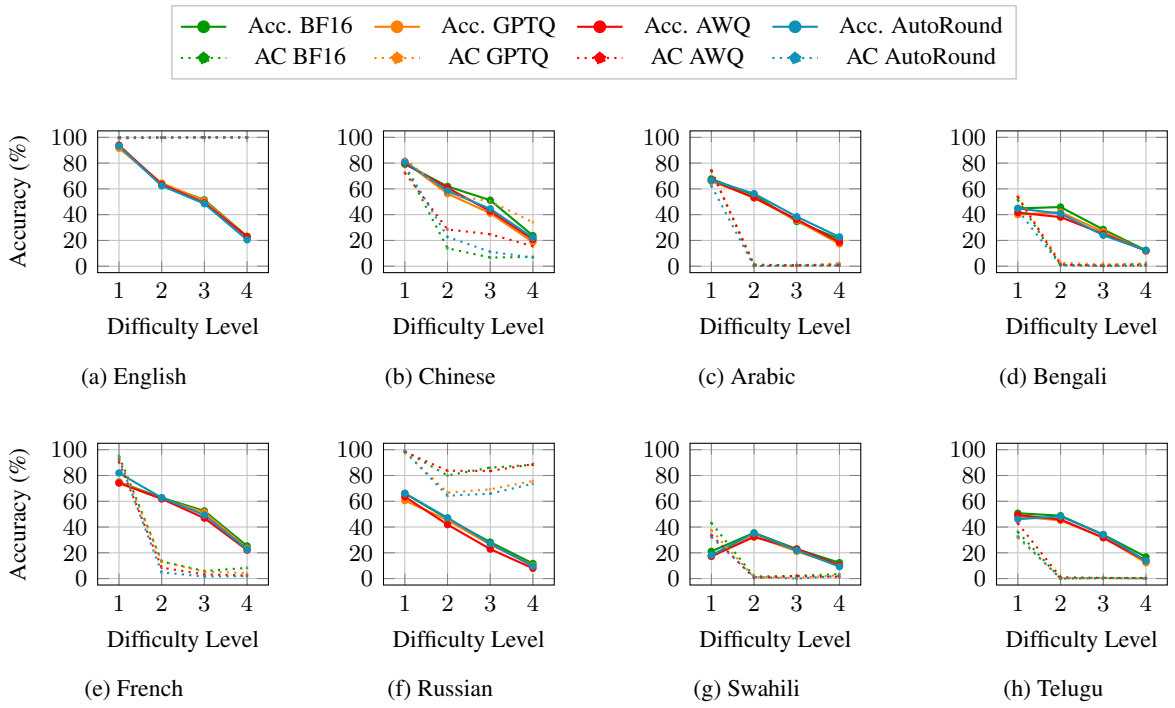

Figure 10: Difficulty-wise accuracy and answer-language consistency (AC) of OLMo-3-7B-Think across eight languages. Solid lines show accuracy while dotted lines show AC across four difficulty levels: 1 = low, 2 = medium, 3 = high, and 4 = top.