Quantized Reasoning Models Think They Need to Think Longer, but They Do Not

Sanae Lotfi
FAIR at Meta
sanaelotfi@meta.com

Polina Kirichenko
FAIR at Meta

Steven Li
Meta AI

Zechun Liu
Meta AI

Abstract

Post-training quantization (PTQ) is widely used to deploy large language models efficiently, but its effect on reasoning models is not well understood. Across math, coding, and science QA, we find that aggressive PTQ reduces accuracy while increasing chain-of-thought (CoT) length. Surprisingly, we show that in up to 52% of the quantized models' failures, models reach the right answer in intermediate reasoning steps but do not output it as a final answer. To understand why quantization leads to this increase in overthinking errors, we measure the token-level KL divergence between quantized and full-precision output distributions. Positions with high KL divergence correlate strongly with high next-token entropy, and at these positions quantized models disproportionately sample overthinking markers such as "wait", "but", and "alternatively". We show that simply introducing a training-free logit penalty on a curated set of overthinking markers can reduce CoT length by 12–23% while preserving or improving accuracy across 5 models (1.5B–32B parameters), 3 quantization methods, and 5 benchmarks, yielding a favorable Pareto frontier of accuracy against reasoning cost compared to penalizing other token sets. Overthinking errors produced by quantized models are particularly reduced by up to 58%.

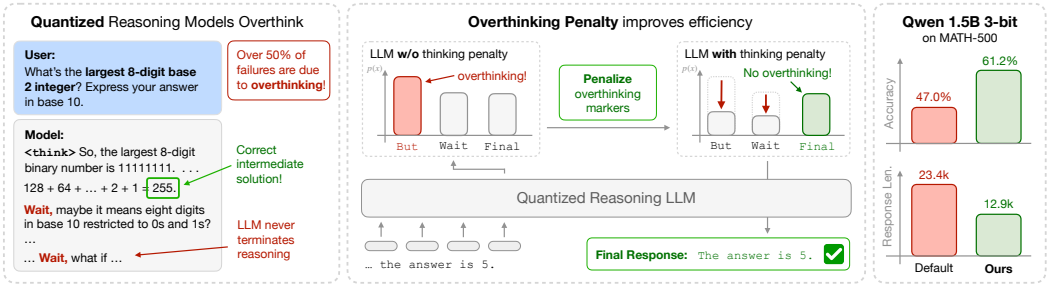


Figure 1: **Quantization exacerbates overthinking in reasoning models. (Left):** Even though quantized reasoning models reach an intermediate correct answer, they self-doubt, open too many new reasoning branches and never produce the final response. **(Middle):** We apply a logit penalty on thinking markers (i.e. "but" or "wait") which open new reasoning branches. **(Right):** We show that the overthinking penalty leads to significant improvements in efficiency (reasoning length) while maintaining or improving accuracy on average across models, datasets and quantization methods.

1 Introduction

Post-training quantization (PTQ) is a widely adopted paradigm for efficient deployment of large language models [1–4]: it requires no gradient updates, no access to the training pipeline, and can be

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

applied with limited calibration data. When a quantized reasoning model fails to solve a problem, the natural assumption is that quantization has eroded the model's capability and the model simply can no longer find the correct answer. In this work, we find that in up to 52% of failures, the quantized model does produce the correct answer at some intermediate reasoning step, but then abandons it. Rather than committing to a correct conclusion, the model opens new reasoning branches, questions its own assumptions, and spirals into redundant deliberation that displaces the answer it had already found. Quantized reasoning models do not fail because they cannot think; they fail because they cannot *stop* thinking.

This failure mode, which we refer to as **overthinking**, emerges as a consistent pattern in post-training quantized (PTQ) reasoning models. While overthinking errors are also present in full-precision models, quantization substantially amplifies their frequency: overthinking accounts for 52% of errors under AWQ 3-bit compared to 26% in BF16, **a** $7.3\times$ **increase in the absolute number of overthinking errors**. Motivated by this gap, we treat the full-precision model as a reference and analyze the token-level KL divergence between quantized and full-precision output distributions under identical generation prefixes to isolate the effects of quantization.

Two key findings emerge. (1) The tokens with the highest average KL divergence between quantized and full-precision models contain *overthinking markers* such as "Wait," "But," and "Alternatively". The tokens with the lowest KL divergence are mathematical and formatting tokens which encode the computational content of reasoning. The position-level KL divergence also correlates strongly with the next-token entropy of the full-precision model, confirming that quantization most affects positions where the model is already uncertain. (2) Overthinking markers concentrate at high-entropy decoding positions as they appear 2–$4\times$ more frequently among the top-20 predictions at high-entropy steps than at low-entropy ones. Because these positions are inherently uncertain, quantization noise has a larger effect on the output distribution, further increasing the likelihood of sampling such tokens. This, in turn, triggers spurious reasoning branches that overwrite correct intermediate answers and delay termination.

To validate this diagnosis, we apply a simple training-free intervention: at each decoding step, a fixed logit penalty is applied to a curated token set $\mathcal{S}$. The penalty introduces zero computational overhead and only a single hyperparameter, yet is highly targeted. The selected tokens correspond to overthinking markers, which are strongly associated with high KL-divergence and high-entropy positions and lexically correspond to hesitation and backtracking. Across all configurations, the penalty consistently reduces CoT length by 12–23% on average. Accuracy is preserved or often improved, reflecting a reduction in overthinking errors. In a few cases, accuracy slightly decreases, but CoT length is still significantly reduced, yielding a favorable shift in the efficiency–performance Pareto frontier. Controlled ablations further highlight the specificity of the intervention: penalizing randomly selected tokens produces no consistent effect, while penalizing tokens with the lowest KL divergence leads to catastrophic degradation where the CoT length increases by up to 41% and the accuracy drops by up to 9.5%. This specificity reflects how quantization degrades reasoning by amplifying path-forking tokens precisely at the high-uncertainty positions where they are most likely to induce repeated branching.

Our contributions are as follows: **(1)** We show that PTQ exacerbates overthinking in reasoning models. In up to 52% of failures under aggressive quantization, the model reaches the correct answer in intermediate steps but fails to commit to it. **(2)** We identify which tokens drive this failure through a KL divergence analysis between quantized and full-precision output distributions. Among the tokens with the highest divergence are hesitation and branching tokens that concentrate at high-entropy positions. The tokens with the lowest divergence are mathematical tokens that carry the computational content of reasoning. A simplified local perturbation analysis provides intuition for why these positions are most affected. **(3)** We validate this diagnosis with a training-free logit penalty on a set of overthinking markers. The penalty consistently reduces CoT length by 12–23% on average while preserving or improving accuracy. Controlled ablations confirm that penalizing overthinking markers yields the most favorable Pareto frontier of accuracy against reasoning cost.

2 Related Work

Post-training quantization for LLMs. Post-training quantization can substantially reduce the model footprint while preserving perplexity and downstream performance. Early work demonstrated robust

8-bit inference for large transformer models via outlier-aware mixed-precision strategies [5]. More recent PTQ methods target 4-bit and 3-bit weights for transformer blocks, including GPTQ [2], which performs layerwise quantization using a second-order approximation of the loss landscape to correct for quantization errors, and AWQ [1], which reduces quantization distortion by identifying and conserving important weight channels using activation statistics from calibration data. Other methods migrate the quantization difficulty from activations to weights such as SmoothQuant [3], or are optimized to reduce quantization and dequantization overhead such as ZeroQuant [4].

Beyond these methods, several recent works improve low-bit PTQ via iterative error corrections. QuIP and QuIP# propose principled low-bit quantization objectives to preserve accuracy at 2–4 bits [6, 7]. Rotation-based approaches reduce outliers and improve quantization, including QuaRot [8] and SpinQuant [9]. FlatQuant enhances the flatness of weights and activations using optimal affine transformations including rotation and per-channel scaling [10]. In our experiments, we focus on 3-bit and 4-bit weight-only quantization using AWQ and GPTQ, as well as 4-bit and 8-bit weight, activation, and KV-cache quantization using FlatQuant.

Efficient reasoning. Overthinking in reasoning models is now a well-documented phenomenon. Chen et al. [11] first quantified overthinking in o1-like LLMs, showing that models systematically over-allocate compute for simple problems, and proposed self-training strategies that reduce token usage by nearly 45% without sacrificing accuracy. Pipis et al. [12] identify repetitive looping as a failure mode, particularly in smaller and distilled models. Su et al. [13] show that the relationship between reasoning length and correctness is non-monotonic: longer traces do not always lead to better answers. Mitigation methods range from fine-tuning models to self-brake before generating redundant steps [14] to dynamic early exit [15] and preferring the shortest correct chain among multiple samples [16]. We refer to Sui et al. [17] for a survey. Our approach differs in being focused on understanding how quantization worsens this overthinking behavior and how to mitigate it without any training, since PTQ itself is training-free.

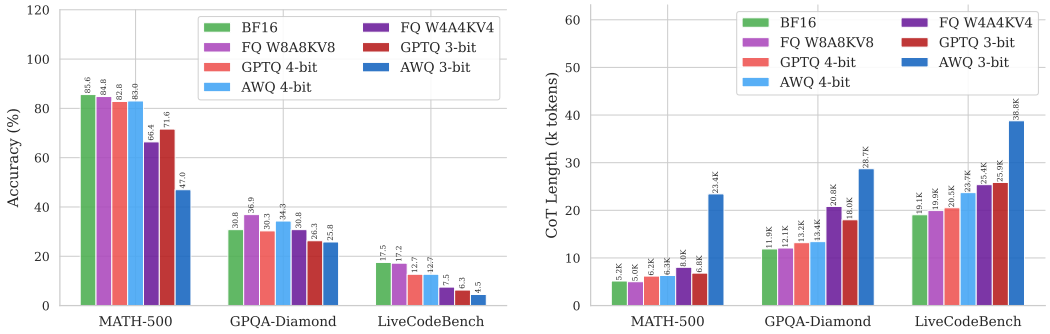
Effect of quantization on reasoning. Recent work suggests that the effect of quantization is metric- and task-dependent, and that compression can preserve aggregate performance while disproportionately degrading harder examples and capabilities [18–20]. Several recent studies report that reasoning models degrade sharply under low-bit PTQ, exhibiting a drop in accuracy and an increase in chain-of-thought length [21–23]. Our work is aligned with this emerging thread but differs in focus. Rather than only benchmarking accuracy under PTQ, we study how quantization changes reasoning traces at the token level, identifying an increase in overthinking errors and introducing a training-free intervention that addresses it.

3 Experimental Setup

Quantization methods. We evaluate post-training quantization under two settings. For weight-only quantization, we use GPTQ [2] and AWQ [1] at 3-bit and 4-bit precision with group size $g{=}128$. GPTQ performs layer-wise quantization using second-order information to minimize output reconstruction error. AWQ identifies salient weight channels using calibration activation statistics and applies per-channel rescaling to reduce quantization distortion. For end-to-end quantization of weights, activations, and the KV cache, we use FlatQuant [10] at W4A4KV4 and W8A8KV8 configurations. We use the notation WbAcKVd to denote b-bit weights, c-bit activations, and d-bit KV cache. Additional details are provided in Section A.

Models. We evaluate five reasoning-specialized LLMs spanning 1.5B to 32B parameters. From the DeepSeek-R1-Distill family, we use DeepSeek-R1-Distill-Qwen 1.5B, 7B, and 14B, as well as DeepSeek-R1-Distill-Llama 8B. These models are distilled from DeepSeek-R1 into Qwen-2.5 and Llama-3.1 architectures, respectively [24]. We additionally evaluate QwQ-32B [25], which acquires reasoning capabilities through reinforcement learning with verifiable rewards rather than distillation.

Benchmarks. We evaluate on five benchmarks spanning mathematics, coding, and science. For mathematics, we use GSM8K [26] for shorter arithmetic chains, MATH-500 [27] for competition-style multi-step problems, and AIME-90 [28] as a harder subset that tends to induce longer reasoning traces. For coding, we use LiveCodeBench [29]. For science, we use GPQA-Diamond [30]. All experiments use default decoding with $T{=}0.6$ and top-$p{=}0.95$, following the setting used by prior work on quantization of reasoning models [21].



(a) Accuracy drops under PTQ

(b) CoT length increases under PTQ

Figure 2: **Quantized reasoning models produce longer chains of thought while achieving lower accuracy.** We evaluate DeepSeek-R1-Distill-Qwen-1.5B in full precision (BF16) and after quantization on MATH-500, GPQA-Diamond, and LiveCodeBench. **(a)** Accuracy decreases as quantization becomes more aggressive, with 3-bit AWQ dropping MATH-500 accuracy from 85.6% to 47.0%. **(b)** CoT length increases significantly under aggressive quantization: 3-bit AWQ increases MATH-500 CoT from 5.2K to 23.4K tokens, a $4.5\times$ increase. Results for all five models are in Section B.

4 Quantization Exacerbates Overthinking in Reasoning Models

4.1 Accuracy drops while CoT length increases under PTQ

First, we study how PTQ affects both accuracy and CoT length on reasoning tasks. Figure 2 shows the effect of progressively more aggressive quantization on DeepSeek-R1-Distill-Qwen-1.5B across three benchmarks. Mild quantization (FlatQuant W8A8KV8 and 4-bit weight-only AWQ and GPTQ) largely preserves accuracy, with CoT lengths remaining close to the BF16 baseline. As precision decreases further, both accuracy and reasoning efficiency degrade together. On MATH-500, 3-bit AWQ reduces accuracy from 85.6% to 47.0% while increasing the average CoT from 5.2K to 23.4K tokens, a $4.5\times$ increase. GPTQ at 3-bit is more robust on the same dataset, retaining 71.6% accuracy with 6.8K tokens, but still produces 32% longer reasoning traces than BF16. FlatQuant W4A4KV4, which quantizes weights, activations, and the KV cache, falls between the two: accuracy drops to 66.4% and CoT grows to 8.0K tokens.

We observe the same pattern across four additional models (Qwen-7B, Qwen-14B, Llama-8B, QwQ-32B) and report full results in Section B. Aggressive quantization (3-bit weight-only and W4A4KV4) consistently leads to longer reasoning traces, with the effect most pronounced for smaller models and harder benchmarks. On LiveCodeBench, 3-bit AWQ increases Qwen-1.5B CoT from 19.1K to 38.8K tokens while accuracy drops from 17.5% to 4.5%. Larger models are more resilient: Qwen-14B under 3-bit AWQ loses only 4.0% average accuracy, and its CoT increase is modest by comparison.

We compute the Spearman correlation coefficient ρ between accuracy degradation and CoT length increase across all 28 model-quantization pairs and find $\rho = -0.73$. In other words, quantized models with the largest accuracy loss tend to also generate the longest reasoning traces. Given this strong correlation, we hypothesize that the additional reasoning induced by quantization is not just a co-occurring effect of lower accuracy, but that the longer chains of thought actively contribute to the performance degradation. We investigate this hypothesis next.

4.2 Quantization increases overthinking errors

To understand what kind of errors quantization leads to, we categorize incorrect answers from DeepSeek-R1-Distill-Qwen-1.5B on MATH-500. We first annotate a subset of errors manually, then use GPT-5 [31] as an LLM judge to scale the categorization after tuning the prompt to achieve over 95% agreement with human annotation. The prompt is provided in Section E.

We assign each failure to exactly one of the following categories: (i) **Overthinking.** The model reaches the correct solution at some point in its chain of thought but does not commit to it as the final answer. Instead, the model opens new reasoning paths, excessively questions its own assumptions,

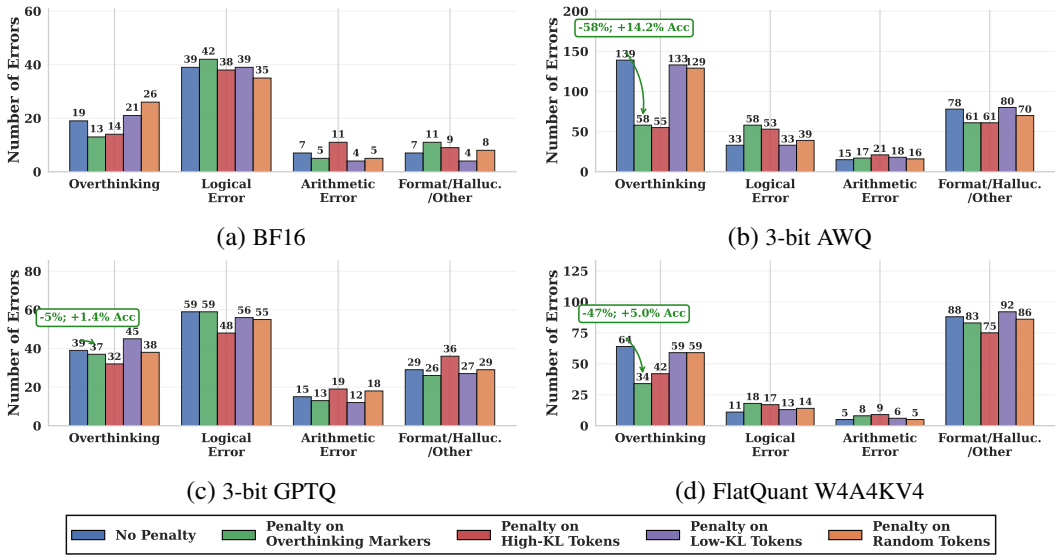


Figure 3: **Quantization increases overthinking errors, and penalizing overthinking markers reduces them.** We categorize errors from DeepSeek-R1-Distill-Qwen-1.5B on MATH-500 into four types. Each panel shows the error count under no penalty and four penalty configurations. Overthinking errors increase under quantization. Penalizing overthinking markers and high-KL tokens consistently reduces overthinking errors by up to 58% on AWQ 3-bit while improving accuracy by up to 14.2%. Penalizing random or low-KL tokens does not reduce overthinking errors and can even increase such errors.

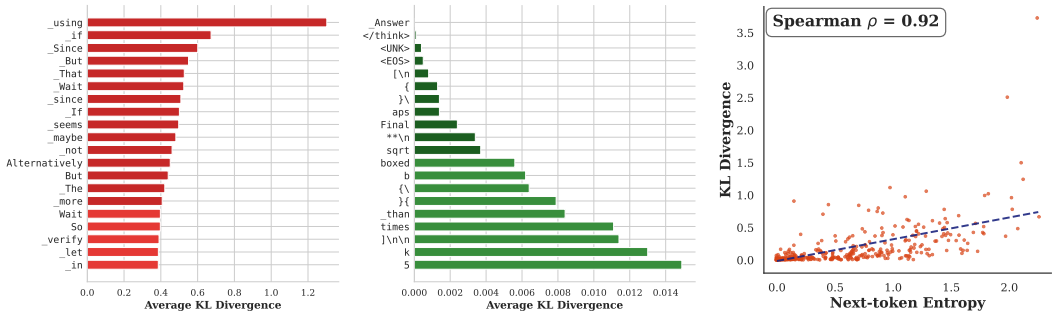
or reverses a correct conclusion. A representative example is shown in Figure 6; (ii) **Logical error.** The model follows an incorrect plan or misunderstands the problem from early on, such that the reasoning trajectory is wrong before any correct solution is reached; (iii) **Arithmetic error.** The overall approach and plan are correct, but the model makes concrete computational or calculation mistakes that lead to a wrong final answer; and (iv) **Formatting, hallucination, and other errors.** The model produces an off-format answer, introduces hallucinated constraints not present in the problem, or exhibits other failures not covered by the categories above.

Figure 3 shows the error breakdown for BF16 and three aggressive quantization methods. In BF16, the model makes 72 errors out of 500 questions, with only 19 attributed to overthinking (26%). Quantization increases overthinking errors across all three methods. Under 3-bit AWQ, overthinking errors rise from 19 to 139, a 7.3× increase, making overthinking the dominant failure mode at 52% of all 265 errors. GPTQ at 3-bit doubles the overthinking count from 19 to 39. FlatQuant more than triples it from 19 to 64. In all three cases, quantization results in an increase in errors where the model reaches the correct answer but fails to commit to it. We show similar results for GSM8K in Figure 10.

This increase in overthinking errors under quantization supports our hypothesis from Section 4.1. The longer reasoning traces produced by quantized models contain excessive hesitation and self-doubt that often turn correct intermediate answers into incorrect final answers.

4.3 Quantized models diverge from non-quantized ones in branching positions

To understand *how* quantization increases overthinking errors, we measure the token-level KL divergence between the output distributions of the BF16 and 3-bit AWQ models. We treat the full-precision model as a reference and run both models on the same MATH-500 prompts under identical generation prefixes to isolate the effects of quantization. At each decoding position t, we compute the KL divergence $D_{\mathrm{KL}}(p_t \| q_t) = \sum_{v \in \mathcal{V}} p_t(v) \log \frac{p_t(v)}{q_t(v)}$, where p_t and q_t are the next-token distributions of the full-precision and quantized models, respectively. We then associate each KL value with the token that the quantized model sampled at that position and compute, for each unique token in the vocabulary, its average KL divergence across all positions where it was sampled. We filter to tokens appearing at least 50 times across all generations.



(a) Highest KL tokens (b) Lowest KL tokens (c) KL vs next-token entropy

Figure 4: **Quantization disproportionately affects thinking and hesitation tokens at high-entropy positions.** We compute the average KL divergence between the BF16 and 3-bit AWQ output distributions for each token appearing at least 50 times on MATH-500. (a) The 20 highest-KL tokens are thinking and branching tokens. (b) The 20 lowest-KL tokens are mathematical and formatting tokens. Tokens prefixed with "_" include a leading space in the tokenizer. (c) Position-level KL divergence correlates strongly with the BF16 model's next-token entropy (Spearman $\rho = 0.92$), confirming that quantization most affects positions where the model is already uncertain.

Figure 4 shows the 20 tokens with the highest and lowest associated average KL divergence. The tokens where quantized and full-precision models diverge the most are thinking and branching tokens such as "Wait", "But", "Alternatively", "if", "maybe", "verify", "think". Many of these tokens open new reasoning paths and express the model's hesitation. The tokens where the two models agree most closely are mathematical and formatting tokens. Quantization barely changes how the model samples these execution tokens. We provide word cloud visualizations of the full set of 50 highest-KL and 50 lowest-KL tokens in Section B.3.

We also compute the density of high-KL and low-KL tokens relative to total CoT length. In BF16, low-KL math tokens appear at a higher rate than high-KL hesitation tokens, with a ratio of 0.57. Under 3-bit AWQ, this ratio flips to 1.15: high-KL tokens outnumber low-KL tokens, meaning the quantized model produces more thinking and hesitation tokens per unit of reasoning than mathematical content. This shift in token composition explains how quantization leads to less efficient reasoning.

5 Penalizing Overthinking Markers

The analysis in Section 4.3 shows that quantization disproportionately affects tokens associated with hesitation and redirection, while leaving mathematical tokens largely unchanged. We now test a simple inference-time intervention that reduces the probability of these tokens during decoding.

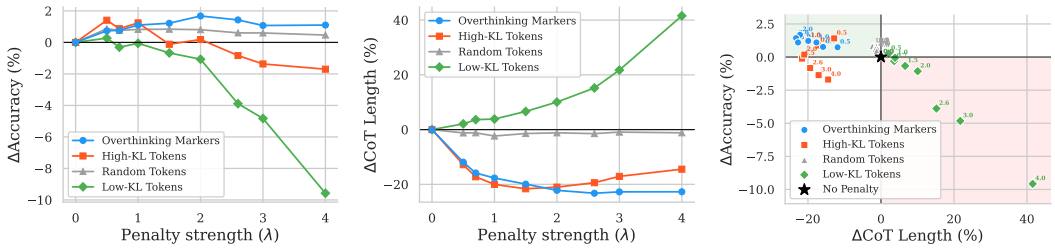
5.1 Overthinking penalty

We curate a set $\mathcal{S}$ of 50 *overthinking markers*: tokens that express hesitation, self-doubt, or reasoning redirection. These include tokens such as "Wait", "But", "Alternatively", "perhaps", "maybe", "however", "reconsider", "backtrack", and "wrong". The full list is provided in Section D.

We construct $\mathcal{S}$ by selecting tokens that signal the model is opening a new reasoning branch or questioning a conclusion it has already reached. Several recent works have identified similar tokens as indicators of redundant reasoning in language models [14, 32]. We deliberately exclude tokens from the high-KL list analyzed in Section 4.3 that serve roles unrelated to overthinking, such as "so" which would in fact help the model conclude its reasoning.

At each decoding step t, we modify the logits $z_t(v)$ for each token $v \in \mathcal{S}$ in our overthinking markers list by subtracting a fixed penalty $\lambda > 0$:

$$z'_t(v) = \begin{cases} z_t(v) - \lambda & \text{if } v \in \mathcal{S}, \\ z_t(v) & \text{otherwise.} \end{cases} \tag{1}$$



(a) Accuracy vs λ (b) CoT length vs λ (c) Pareto frontier

Figure 5: **Penalizing overthinking markers consistently reduces CoT length while preserving accuracy.** We compare four token lists (50 tokens each) for the logit penalties on Qwen-1.5B, averaged across six quantization configurations and five benchmarks. (a) Penalizing overthinking markers maintains or improves accuracy across all λ values, while penalizing low-KL tokens degrades accuracy. (b) Overthinking markers consistently reduce CoT length by 12% to 23%. Low-KL tokens increase CoT by up to 41%. Penalizing random tokens has negligible effect. (c) Overthinking markers occupy the upper-left region (shorter CoT, better accuracy) across all λ values, yielding the best efficiency-accuracy tradeoff. Black star marks the no-penalty baseline, and penalizing random tokens remains close to this baseline. Note that the numbers on the markers reflect the penalty strength.

The penalty does not require additional forward passes and adds no computational overhead. The only hyperparameter is λ, which controls the strength of the overthinking penalty.

5.2 Penalizing overthinking markers reduces CoT while preserving accuracy on average

We compare the effect of penalizing four different token lists of equal size (50 tokens each): our overthinking markers, the tokens with the highest KL divergence between BF16 and AWQ 3-bit (high-KL tokens), the tokens with the lowest KL divergence (low-KL tokens), and 50 randomly selected tokens. We apply the logit penalty as defined in Equation (1) to each list when evaluating Qwen-1.5B across six quantization configurations and five benchmarks, sweeping λ from 0.5 to 4.0.

Figure 5 shows the results averaged across all configurations and benchmarks. Penalizing overthinking markers consistently reduces CoT length by 12% to 23% across all λ values while maintaining or slightly improving accuracy. Penalizing random tokens has negligible effect on both accuracy and CoT length. Penalizing high-KL tokens reduces CoT at low λ but degrades accuracy at higher values, since this list includes tokens that could actually help terminate the reasoning. Penalizing low-KL tokens is catastrophic: CoT increases by up to 41% and accuracy drops by up to 9.5%, confirming that these tokens are not the ones causing post-quantization degradation.

We further verify the directionality of the penalty by running a symmetric sweep with negative λ values that boost rather than suppress the target tokens as shown in Figure 9. Boosting overthinking markers increases CoT length by up to 445% and drops accuracy by up to 34%. High-KL tokens show a comparable effect, while random tokens remain neutral. These results confirm that overthinking markers and high-KL tokens are the tokens that control the reasoning length in both directions.

We show in Figure 5c that overthinking markers occupy the upper-left region of the Pareto frontier *for all λ values*, achieving shorter CoT without sacrificing accuracy. No other token list achieves this: random tokens cluster near the no-penalty origin, high-KL tokens drift toward worse accuracy at strong penalties, and low-KL tokens move into the lower-right region where both accuracy and efficiency degrade. We also note that the overthinking penalty is robust to the choice of λ: every λ value tested reduces CoT length by at least 12% on average across configurations on Qwen-1.5B while preserving performance as in Figures 5a and 5b.

In Table 1, we report results with the best λ per configuration alongside the standard deviation across all λ values.

Table 1: **Penalizing overthinking markers reduces CoT length significantly while maintaining or improving accuracy on average under extreme quantization.** For each configuration, the top row (Base) shows accuracy (%) and CoT length (thousands of tokens) without penalty, and the bottom row (+Pen) shows results with the best λ; we report the standard deviation w.r.t λ in the Δ columns. Our overthinking penalty consistently reduces CoT length across all configurations, yielding a better Pareto frontier of accuracy against reasoning cost than the baseline. In several cases accuracy also improves. **Per-benchmark cells are colored when the penalty changes accuracy or CoT length by $\geq$2%.** We show the most aggressive quantization settings; and report the rest in Section B.

			AIME	MATH	GSM8K	GPQA	LCB	Avg.	ΔAcc	ΔLen%
			Acc (%) / Len (k)					*Acc / Len*	*mean$_{\pm std}$*	
Qwen-1.5B	**BF16**	Base	**24.4 / 17.5**	**85.6 / 5.2**	**84.2 / 2.7**	**30.8 / 11.9**	**17.5 / 19.1**	**48.5 / 11.2**		
		+Pen	25.6 / 14.8	85.8 / 4.5	85.1 / 2.6	39.4 / 10.5	17.5 / 18.3	50.7 / 10.1	$+2.1_{\pm 1.6}$	$-9.2_{\pm 5.7}$
	GPTQ W3	Base	8.9 / 20.5	71.6 / 6.8	76.1 / 4.0	26.3 / 18.0	6.3 / 25.9	37.8 / 15.0		
		+Pen	11.1 / 15.6	73.0 / 5.8	77.5 / 2.8	28.8 / 12.5	8.2 / 22.2	39.7 / 11.8	$+1.9_{\pm 1.1}$	$-22.6_{\pm 6.2}$
	AWQ W3	Base	4.4 / 33.0	47.0 / 23.4	63.5 / 16.3	25.8 / 28.7	4.5 / 38.8	29.0 / 28.0		
		+Pen	8.9 / 29.3	61.2 / 12.9	69.5 / 7.6	30.3 / 22.7	6.3 / 35.3	35.2 / 21.6	$+6.2_{\pm 1.7}$	$-28.0_{\pm 7.5}$
	FQ W4A4KV4	Base	7.8 / 17.5	66.4 / 8.0	78.5 / 3.4	30.8 / 20.8	7.5 / 25.4	38.2 / 15.0		
		+Pen	12.2 / 16.8	71.4 / 6.2	78.9 / 2.1	31.3 / 17.6	8.6 / 23.8	40.5 / 13.3	$+2.3_{\pm 1.3}$	$-17.6_{\pm 4.9}$
Qwen-7B	**BF16**	Base	**45.6 / 12.7**	**93.2 / 3.9**	**91.1 / 1.9**	**50.0 / 9.9**	**37.7 / 16.5**	**63.5 / 9.0**		
		+Pen	52.2 / 11.2	94.2 / 3.5	91.6 / 1.6	56.6 / 9.4	38.8 / 14.9	66.7 / 8.1	$+3.2_{\pm 1.9}$	$-10.0_{\pm 2.7}$
	GPTQ W3	Base	30.0 / 17.0	91.0 / 4.8	90.1 / 2.5	41.9 / 11.4	27.2 / 19.1	56.0 / 11.0		
		+Pen	34.4 / 14.8	91.2 / 4.2	90.8 / 1.8	48.5 / 10.3	26.5 / 19.2	58.3 / 10.1	$+2.2_{\pm 1.5}$	$-13.1_{\pm 3.0}$
	AWQ W3	Base	30.0 / 16.1	91.2 / 4.8	90.4 / 2.1	48.0 / 10.7	26.9 / 21.3	57.3 / 11.0		
		+Pen	28.9 / 14.6	90.2 / 4.0	90.5 / 1.8	49.0 / 10.1	28.7 / 18.9	57.5 / 9.9	$+0.2_{\pm 1.4}$	$-12.0_{\pm 3.9}$
	FQ W4A4KV4	Base	23.3 / 20.4	83.6 / 5.4	87.7 / 2.1	50.0 / 18.0	12.3 / 23.9	51.4 / 14.0		
		+Pen	22.2 / 19.7	86.4 / 4.7	88.5 / 2.1	51.5 / 14.9	14.2 / 23.2	52.6 / 12.9	$+1.2_{\pm 2.4}$	$-7.7_{\pm 4.8}$
Qwen-14B	**BF16**	Base	**57.8 / 13.1**	**94.6 / 3.7**	**94.1 / 1.7**	**57.6 / 9.5**	**49.6 / 14.4**	**70.7 / 8.5**		
		+Pen	63.3 / 10.7	95.6 / 3.5	94.5 / 1.7	62.6 / 8.8	54.5 / 13.4	74.1 / 7.6	$+3.4_{\pm 1.7}$	$-8.5_{\pm 2.3}$
	GPTQ W3	Base	44.4 / 12.4	94.4 / 3.6	92.8 / 1.6	53.5 / 9.1	42.2 / 14.6	65.5 / 8.3		
		+Pen	47.8 / 11.9	93.8 / 3.4	92.5 / 1.5	56.1 / 8.7	47.4 / 14.6	67.5 / 8.0	$+2.0_{\pm 1.7}$	$-4.8_{\pm 2.3}$
	AWQ W3	Base	48.9 / 14.3	93.0 / 4.3	93.0 / 2.1	56.6 / 10.5	42.2 / 17.5	66.7 / 9.7		
		+Pen	47.8 / 14.1	94.2 / 3.9	92.9 / 1.7	56.1 / 10.2	45.9 / 16.2	67.4 / 9.2	$+0.6_{\pm 1.7}$	$-7.5_{\pm 3.5}$
	FQ W4A4KV4	Base	52.2 / 12.7	95.0 / 3.7	93.5 / 1.8	63.6 / 8.8	49.3 / 15.2	70.7 / 8.4		
		+Pen	53.3 / 12.6	94.6 / 3.6	93.9 / 1.6	61.6 / 8.8	49.3 / 14.3	70.5 / 8.2	$-0.2_{\pm 1.3}$	$-4.5_{\pm 3.5}$
Llama-8B	**BF16**	Base	**44.4 / 14.5**	**89.0 / 4.6**	**89.0 / 2.2**	**49.0 / 10.2**	**35.1 / 16.2**	**61.3 / 9.6**		
		+Pen	42.2 / 13.8	91.0 / 4.2	88.6 / 2.0	50.5 / 10.1	38.4 / 15.3	62.1 / 9.1	$+0.8_{\pm 2.0}$	$-5.6_{\pm 2.7}$
	GPTQ W3	Base	26.7 / 18.2	82.2 / 6.2	85.4 / 2.4	29.8 / 12.8	24.6 / 24.1	49.7 / 12.7		
		+Pen	28.9 / 16.2	83.4 / 5.6	85.8 / 2.0	39.9 / 12.7	26.9 / 23.0	53.0 / 11.9	$+3.2_{\pm 1.9}$	$-7.9_{\pm 3.9}$
	AWQ W3	Base	12.2 / 14.8	80.4 / 5.4	82.6 / 2.4	35.9 / 9.6	23.1 / 18.1	46.8 / 10.1		
		+Pen	16.7 / 13.6	77.6 / 4.7	83.7 / 2.0	36.9 / 8.6	23.1 / 15.9	47.6 / 9.0	$+0.7_{\pm 1.5}$	$-12.2_{\pm 2.7}$
	FQ W4A4KV4	Base	22.2 / 15.4	86.4 / 5.0	87.6 / 2.4	40.9 / 10.2	35.1 / 17.8	54.4 / 10.2		
		+Pen	22.2 / 14.7	84.2 / 4.9	86.1 / 2.2	48.5 / 9.8	31.7 / 17.3	54.5 / 9.8	$+0.1_{\pm 1.7}$	$-4.1_{\pm 3.0}$
QwQ-32B	**BF16**	Base	**76.7 / 15.3**	**97.2 / 4.5**	**95.7 / 1.9**	**63.1 / 10.1**	**62.3 / 17.9**	**79.0 / 9.9**		
		+Pen	81.1 / 12.3	98.0 / 4.0	96.0 / 1.8	66.7 / 9.5	61.9 / 17.3	80.7 / 9.0	$+1.7_{\pm 1.2}$	$-9.6_{\pm 2.6}$
	GPTQ W3	Base	64.4 / 16.1	96.8 / 4.9	95.4 / 2.0	56.1 / 9.3	50.4 / 19.1	72.6 / 10.3		
		+Pen	65.6 / 15.0	96.4 / 4.3	95.9 / 1.8	60.1 / 9.5	51.9 / 18.0	74.0 / 9.7	$+1.3_{\pm 1.4}$	$-5.9_{\pm 3.9}$
	AWQ W3	Base	60.0 / 16.7	96.0 / 4.6	95.9 / 1.8	61.1 / 10.1	51.5 / 19.5	72.9 / 10.5		
		+Pen	62.2 / 14.5	96.6 / 4.2	95.5 / 1.7	60.6 / 8.7	53.4 / 18.7	73.7 / 9.5	$+0.7_{\pm 1.2}$	$-9.6_{\pm 2.9}$

5.3 Overthinking penalty improves efficiency across models

Table 1 shows the effect of the overthinking penalty across five models and the most aggressive quantization settings. The penalty reduces CoT length by 4.1% to 28.0% on average across benchmarks. In most cases, accuracy is preserved or slightly improved. Notably, there are a few cases where the accuracy deteriorates but the CoT length improves. On AWQ 3-bit with Qwen-1.5B, accuracy improves by 6.2% while CoT decreases by 28.0%. Larger models show smaller but consistent CoT reductions, improving the efficiency of quantized models and partially reversing the overthinking behavior exacerbated by quantization. We report full results for all quantization methods in Section B.

Figure 3 shows that the penalty on overthinking markers reduces the number of cases where the model reaches the correct answer but fails to commit. On AWQ 3-bit, overthinking errors drop from 139 to 58, a 58% reduction, while total errors decrease from 265 to 194. On FlatQuant W4A4KV4,

overthinking errors drop from 64 to 34 (47% reduction) with total errors decreasing from 168 to 143. Penalizing random tokens or low-KL tokens can actually lead to an increase in overthinking errors, demonstrating the effectiveness of the penalty over our curated list of overthinking markers. We observe the same pattern on GSM8K, where penalizing overthinking markers reduces overthinking errors across the three extreme quantization configurations from 344 to 237, a 31% reduction as shown in Section B.5.

The overthinking penalty also reduces CoT length for BF16 models, with average reductions between 5.6% and 10.0% depending on the model. Quantization does not introduce overthinking as a new failure mode but rather exacerbates a tendency already present in full-precision models, as reflected by both the CoT increase after quantization and the increase of overthinking errors in particular.

6 Why Does Quantization Exacerbate Overthinking?

In the previous sections we demonstrated that quantization increases overthinking errors and CoT length, and that penalizing overthinking markers addresses both. We now discuss, formalize, and empirically verify a hypothesis for *why* quantization worsens overthinking in reasoning models.

Yayla and Kumar [33] show that the robustness of a quantized neural network to parameter perturbations is governed by the output-layer logit margin: the gap $m = z_{(1)} - z_{(2)}$ between the top-1 and top-2 logits. When this margin is large, a perturbation to the model's weights is unlikely to flip the argmax; when the margin is small, even a moderate perturbation can change the selected output. While their analysis targets classification networks under deterministic bit-flip errors, the same structural argument applies to the autoregressive setting under stochastic quantization noise [34, 35]: locally uncertain positions can be affected more aggressively by quantization noise.

Local perturbation analysis. We provide a local perturbation analysis to formalize the intuition behind the effect of PTQ. Let us denote the next-token distribution as $p_t = \mathrm{softmax}(z_t)$ and approximate quantization locally as isotropic Gaussian logit noise, $z_t^{\text{quant}} = z_t + \epsilon_t$, with $\epsilon_t \sim \mathcal{N}(0, \sigma_q^2 I)$, so that $q_t = \mathrm{softmax}(z_t^{\text{quant}})$. The expected KL divergence is

$$\mathbb{E}_{\epsilon_t}\big[D_{\mathrm{KL}}(p_t \,\|\, q_t)\big] \approx \frac{\sigma_q^2}{2}\Big(1 - \sum_{v \in \mathcal{V}} p_t(v)^2\Big). \tag{2}$$

This suggests that the expected shift is small when one next token dominates and larger when probability mass is diffuse. The factor $1 - \sum_v p_t(v)^2$ is monotone in the Rényi-2 entropy, and we show below that entropy is a strong empirical proxy for the observed shift.

The same noise model also provides an argmax-stability intuition. Let $m_t = z_{t,(1)} - z_{t,(2)}$ be the gap between the two largest logits, and let Φ be the standard normal CDF. The probability that the runner-up overtakes the top token gives the lower bound $P(\text{flip}) \geq \Phi\big(-m_t/(\sqrt{2}\sigma_q)\big)$. A union bound over all competing tokens gives the upper bound $P(\text{flip}) \leq (|\mathcal{V}|-1)\,\Phi\big(-m_t/(\sqrt{2}\sigma_q)\big)$. Both bounds decrease with the logit gap, suggesting that low-margin positions are more vulnerable under this approximation. We provide the derivations in Section C.

We verify this intuition in the autoregressive setting. We show in Figure 4c that the correlation coefficient between the KL divergence computed on MATH-500 under identical generation prefixes and the next-token entropy of the full-precision model is $\rho = 0.92$ across all positions, confirming that positions where the model is most uncertain (small logit gap, high entropy) are where the quantized and full-precision models diverge the most.

The connection to overthinking follows from a second empirical observation: overthinking markers are 2 to $4\times$ more likely to appear among the top-20 most probable tokens at high-entropy positions compared to low-entropy positions as shown in Figure 7. At these uncertain positions, quantization noise can shift the sampled token toward an overthinking marker such as "Wait" or "But", causing the model to open a new reasoning branch instead of continuing its current path. In summary, under this very simplified model, quantization noise raises $P(\text{flip})$ throughout decoding, but the increase is larger at diffuse, low-margin positions. These are the high-entropy positions where hesitation markers are empirically more likely to appear. Quantization therefore produces larger observed distribution shifts for hesitation tokens because of where they occur in the decoding trajectory, not because quantization noise directly targets their lexical identity. We note that this argument provides

intuition for one specific failure mode, not a complete causal explanation of all types of degradations caused by PTQ.

7 Conclusion, Limitations, and Future Work

In this work, we studied the effect of post-training quantization on reasoning models and found that aggressive quantization exacerbates overthinking, where the model reaches correct intermediate answers but fails to commit to them, leading to longer chains of thought and lower accuracy. Through a token-level KL divergence analysis between quantized and full-precision models, we identified that hesitation and branching tokens are disproportionately affected by quantization while mathematical tokens remain stable. A simple training-free logit penalty on 50 curated overthinking markers consistently reduces CoT length by 12–23% leading to a better accuracy-reasoning cost Pareto frontier across five models, three quantization methods, and five benchmarks.

Overthinking is not unique to quantized models. BF16 models also produce overthinking errors, and the penalty reduces CoT length for full-precision models by 5.6% to 10.0% depending on the model. *Quantization amplifies a failure mode already present in reasoning models*. Our KL divergence analysis is also not specific to quantization: any perturbation to a model's weights or activations, from pruning, distillation, or low-rank approximation, may similarly affect uncertain positions. Comparing token-level divergences between a reference model and its compressed variant could serve as a general diagnostic for identifying which aspects of generation are most vulnerable to compression.

Several directions remain open. Our penalty uses a fixed λ at every decoding step. While the ablation shows robustness across λ values, adapting λ dynamically based on local entropy is a natural extension. The overthinking markers list is manually curated for English reasoning models. Automatically constructing the list per model would make our approach more widely applicable. Finally, our evaluation focuses on math, coding, and science benchmarks. Whether the same overthinking patterns arise in other reasoning and planning domains remains an open question.

References

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

[2] 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.

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

[4] Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in neural information processing systems*, 35:27168–27183, 2022.

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

[6] Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher M De Sa. Quip: 2-bit quantization of large language models with guarantees. *Advances in Neural Information Processing Systems*, 36:4396–4429, 2023.

[7] Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. Quip#: Even better llm quantization with hadamard incoherence and lattice codebooks. *arXiv preprint arXiv:2402.04396*, 2024.

[8] Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: Outlier-free 4-bit

inference in rotated llms. *Advances in Neural Information Processing Systems*, 37:100213–100240, 2024.

[9] Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. Spinquant: Llm quantization with learned rotations. *arXiv preprint arXiv:2405.16406*, 2024.

[10] Yuxuan Sun, Ruikang Liu, Haoli Bai, Han Bao, Kang Zhao, Yuening Li, Jiaxin Hu, Xianzhi Yu, Lu Hou, Chun Yuan, et al. Flatquant: Flatness matters for llm quantization. *arXiv preprint arXiv:2410.09426*, 2024.

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

[12] Charilaos Pipis, Shivam Garg, Vasilis Kontonis, Vaishnavi Shrivastava, Akshay Krishnamurthy, and Dimitris Papailiopoulos. Wait, wait, wait... why do reasoning models loop? *arXiv preprint arXiv:2512.12895*, 2025.

[13] Jinyan Su, Jennifer Healey, Preslav Nakov, and Claire Cardie. Between underthinking and overthinking: An empirical study of reasoning length and correctness in llms. *arXiv preprint arXiv:2505.00127*, 2025.

[14] Haoran Zhao, Yuchen Yan, Yongliang Shen, Haolei Xu, Wenqi Zhang, Kaitao Song, Jian Shao, Weiming Lu, Jun Xiao, and Yueting Zhuang. Let llms break free from overthinking via self-braking tuning. *arXiv preprint arXiv:2505.14604*, 2025.

[15] Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. Dynamic early exit in reasoning models. *arXiv preprint arXiv:2504.15895*, 2025.

[16] Michael Hassid, Gabriel Synnaeve, Yossi Adi, and Roy Schwartz. Don't overthink it. preferring shorter thinking chains for improved llm reasoning. *arXiv preprint arXiv:2505.17813*, 2025.

[17] Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, et al. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

[18] Sara Hooker, Nyalleng Moorosi, Gregory Clark, Samy Bengio, and Emily Denton. Characterising bias in compressed models. *arXiv preprint arXiv:2010.03058*, 2020.

[19] Sanae Lotfi, Yilun Kuang, Marc Finzi, Brandon Amos, Micah Goldblum, and Andrew G Wilson. Unlocking tokens as data points for generalization bounds on larger language models. *Advances in Neural Information Processing Systems*, 37:9229–9256, 2024.

[20] Stanley Z Hua, Sanae Lotfi, and Irene Y Chen. Uncertainty drives social bias changes in quantized large language models. *arXiv preprint arXiv:2602.06181*, 2026.

[21] 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.

[22] Anmol Mekala, Anirudh Atmakuru, Yixiao Song, Marzena Karpinska, and Mohit Iyyer. Does quantization affect models' performance on long-context tasks? *arXiv preprint arXiv:2505.20276*, 2025.

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

[24] Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men,

Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, Shanghaoran Quan, and Zekun Wang. Qwen2.5 technical report. *ArXiv*, abs/2412.15115, 2024. URL `https://api.semanticscholar.org/CorpusID:274859421`.

[25] Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, 2025.

[26] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[27] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

[28] Maxwell-Jia. Aime 2024 dataset. `https://huggingface.co/datasets/Maxwell-Jia/AIME`, 2025.

[29] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

[30] 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.

[31] Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

[32] Chenrui Fan, Ming Li, Lichao Sun, and Tianyi Zhou. Missing premise exacerbates overthinking: Are reasoning models losing critical thinking skill? *arXiv preprint arXiv:2504.06514*, 2025.

[33] Mikail Yayla and Akash Kumar. Mcel: Margin-based cross-entropy loss for error-tolerant quantized neural networks. *arXiv preprint arXiv:2603.05048*, 2026.

[34] Bernard Widrow, Istvan Kollar, and Ming-Chang Liu. Statistical theory of quantization. *IEEE Transactions on instrumentation and measurement*, 45(2):353–361, 1996.

[35] Alexandre Défossez, Yossi Adi, and Gabriel Synnaeve. Differentiable model compression via pseudo quantization noise. *arXiv preprint arXiv:2104.09987*, 2021.

[36] Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9, 2024.

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

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

[39] Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*, 2025.

A Experimental Details

A.1 Calibration Data

GPTQ is calibrated on *reasoning data* to mitigate domain shift: we use model-generated reasoning traces on NuminaMath-1.5 [36], sampling 128 sequences of length 2048. For AWQ, we follow the standard AWQ recipe and sample 128 sequences of length 512 from the Pile uncopyrighted dataset[1] [37, 21]. For FlatQuant, we use 128 text sequences of length 2048 from WikiText2 [38] by default. Following Liu et al. [21], we increase the calibration sequence length to 4096 for DeepSeek-R1-Distill-Qwen-1.5B and 7B to avoid extreme outliers in the K cache.

A.2 Quantization Methods

Asymmetric group-wise quantization. We use group-wise asymmetric quantization with group size $g = 128$ along the input dimension, which is standard in GPTQ and AWQ quantization. For each output channel i and each consecutive group of g weights within that channel, we quantize a weight vector $w \in \mathbb{R}^g$ as $\widehat{w} = s\,(q - z)$, where $q = \mathrm{clip}\big(\lfloor \frac{w}{s} + z \rceil\,,\ 0,\ 2^b - 1\big)$, $b \in \{4, 3\}$ is the bit-width, $s > 0$ is a learned per-group scale, $z \in \mathbb{Z}$ is a per-group zero-point, $\lfloor \cdot \rceil$ denotes rounding to the nearest integer, and clip clamps to the representable integer range.

GPTQ (weight-only, 3/4-bit). GPTQ constructs $\widehat{W}$ by quantizing one layer at a time while using second-order information to compensate for quantization error [2]. Given calibration activations X, the goal is to minimize the output reconstruction error: $\min_{\widetilde{W} \in \mathcal{Q}_b} \quad \|X\widetilde{W}^\top - XW^\top\|_F^2$, where $\mathcal{Q}_b$ denotes the set of matrices representable by the chosen b-bit group-wise quantizer. GPTQ approximately solves this using a blockwise sequential procedure and an approximate inverse-Hessian estimated from calibration data, implemented efficiently via Cholesky-based updates [2]. We use group size $g = 128$ and evaluate 3-bit and 4-bit weight-only GPTQ.

AWQ (weight-only, 3/4-bit). AWQ is *activation-aware* weight-only PTQ: it uses calibration activations to identify salient channels and applies a channel-wise rescaling before quantization to reduce output distortion [1]. For a diagonal scaling matrix S applied to the input features, the linear layer can be rewritten as $XW^\top = (XS)\,(WS^{-1})^\top$. AWQ chooses S from calibration statistics so that the rescaled weights WS^{-1} are easier to quantize while preserving the original function. The rescaling is folded into the stored weights, so AWQ remains weight-only at inference time. We use group size $g = 128$ and evaluate 3-bit and 4-bit weight-only AWQ.

FlatQuant (W4A4KV4 and W8A8KV8). Weight-only PTQ reduces model footprint but does not address other inference bottlenecks that dominate long-form generation, such as activation memory, bandwidth and KV-cache storage. To study these effects, we additionally use FlatQuant [10], which quantizes weights, activations, and the KV cache with lightweight calibration, enabling end-to-end low-precision inference without gradient-based retraining. We evaluate two configurations: W4A4KV4 (4-bit weights, activations, and KV cache) and W8A8KV8 (8-bit weights, activations, and KV cache).

Bit-widths. Our baseline models use BF16 for weights, activations, and the KV cache. For weight-only PTQ, we evaluate $b \in \{4, 3\}$ for both GPTQ and AWQ, matching the low-bit settings studied in prior work on quantized reasoning models [21]. For FlatQuant, we evaluate W4A4KV4 and W8A8KV8.

A.3 Penalty Sweeps

We use temperature T=0.6 and nucleus sampling top-p=0.95 as default decoding hyperparameters throughout all experiments, following previous work [21].

Full model sweep (V1 overthinking markers). We evaluate the overthinking markers penalty (50 tokens, listed in Section D) across all five models and all quantization configurations. We sweep the

[1] `https://huggingface.co/datasets/monology/pile-uncopyrighted`

penalty strength over $\lambda \in \{0.5, 0.7, 1.0, 1.5, 2.0, 2.6, 3.0, 4.0\}$ and evaluate on all five benchmarks. In Table 1, we report the best accuracy after the sweep over λ, the corresponding CoT length, alongside the standard deviation of accuracy and CoT length across all λ values. Figure 5 shows the average change in performance over all datasets and quantization configurations for each value of λ.

Token list ablation (Qwen-1.5B). We compare four token lists of equal size (50 tokens each) on Qwen-1.5B across six quantization configurations (GPTQ W3, GPTQ W4, AWQ W3, AWQ W4, FlatQuant W4A4KV4, FlatQuant W8A8KV8) and five benchmarks. We sweep $\lambda \in \{0.5, 0.7, 1.0, 1.5, 2.0, 2.6, 3.0, 4.0\}$ for each list. The four lists are: overthinking markers (50 hand-curated hesitation and redirection tokens), high-KL tokens (top 50 highest average KL tokens between BF16 and AWQ W3 with frequency ≥ 50), random tokens (50 randomly selected tokens not in the other lists), and low-KL tokens (50 tokens with the lowest average KL divergence with frequency ≥ 50). All lists are provided in Section D. This ablation totals $4 \times 6 \times 5 \times 8 = 960$ penalty runs plus 30 baseline runs. Results are shown in Figure 5.

Negative penalty sweep (Qwen-1.5B). To verify that the direction of the intervention matters, we run an additional sweep with negative λ values symmetric to the positive sweep: $\lambda \in \{-0.5, -0.7, -1.0, -1.5, -2.0, -2.6, -3.0, -4.0\}$. Negative penalties *boost* the logits of the target tokens rather than suppressing them. We run the same four token lists across the same six quantization configurations and five benchmarks on Qwen-1.5B, totaling an additional 960 runs. Results are reported in Section B.

B Additional Results

B.1 Qualitative Example of Overthinking from MATH-500

Figure 6 shows an actual MATH-500 prompt and the corresponding model outputs from DeepSeek-R1-Distill-Qwen-1.5B in BF16 versus the same model quantized to 3-bit with AWQ. While the quantized model initially reaches the correct intermediate result, it then samples repeated thinking/path-forking tokens (e.g., "wait", "but") and branches into alternative reasoning paths, leading to excessive CoT length and a failure to terminate. This is a representative overthinking failure mode example in our error analysis.

B.2 Position of Thinking Tokens in the Next-token Probability Distribution

We analyze where thinking/hesitation tokens appear in the model's next-token probability distribution, and how this changes between high-entropy and low-entropy decoding positions. Concretely, for each generation step t we compute the next-token entropy H_t, rank all vocabulary items by their predicted probability, and record whether at least one token from our thinking-token lexicon (defined in Section D) appears among the top-20 most probable next tokens. We then compare this top-20 inclusion rate at *high-entropy positions* ($H_t > \tau$) versus *low-entropy positions* ($H_t \leq \tau$), where τ is set per model to the 80th percentile of its entropy distribution.

Figure 7 shows that thinking tokens are consistently ≈ 2–$4\times$ more likely to be ranked in the top-20 at high-entropy positions across DeepSeek-R1-Distill-Qwen-1.5B and QwQ-32B, including their 3-bit quantized variants (AWQ and GPTQ). Importantly, these inclusion rates are high despite the very large vocabulary sizes (151,936 tokens for DeepSeek-R1-Distill-Qwen-1.5B and 152,064 tokens for QwQ-32B). High-entropy positions are precisely the positions where thinking tokens become more likely under sampling, making them natural triggers for branching and overthinking.

B.3 Word Cloud Visualizations of High-KL and Low-KL Tokens

We show word cloud visualizations of the 50 tokens with the highest and lowest average KL divergence between the BF16 and 3-bit AWQ output distributions on MATH-500 in Figure 8. Token size is proportional to the average KL divergence. The high-KL word cloud is dominated by hesitation and redirection tokens that open new reasoning branches, while the low-KL word cloud consists of mathematical symbols, digits, and formatting tokens that carry the computational content of the reasoning trace.

? Question

What's the largest eight-digit base 2 integer? Express your answer in base 10. (Please reason step by step, and put your final answer within `\boxed{}`.)

Ideal steps followed by the non-quantized model ≃ 1124 tokens

1. An "eight-digit base 2 integer" means a binary number with exactly 8 digits.
2. The largest such number is 11111111_2.
3. Convert to base 10: $11111111_2 = \sum_{k=0}^{7} 2^k = 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255$
4. Therefore, the largest eight-digit base 2 integer in base 10 is:

$$\boxed{255}$$

vs. 3-bit quantized model: → 5494 tokens

Correct reasoning start:

"So, the largest eight-digit binary number is 11111111_2. Each digit represents a power of two ... $128+64+32+16+8+4+2+1 = 255$." ✓

Consecutive path-forking and overthinking:

"Wait, but maybe 'eight-digit' means something else? ... Wait, maybe it means eight digits in base 10 restricted to 0s and 1s? ... Wait, what if the phrasing is tricky?" ✗

✗ **Model never terminates!**

Figure 6: **Quantization-induced overthinking in a single trace.** We show an actual question from MATH-500 and the responses of DeepSeek-R1-Distill-Qwen-2.5- 1.5B in BF16 versus the same model quantized to 3-bit with AWQ. The BF16 model follows a short, concise solution, while a 3-bit quantized model reaches the correct reasoning early but then injects path-forking "wait"-style tokens that trigger branching and can stall termination.

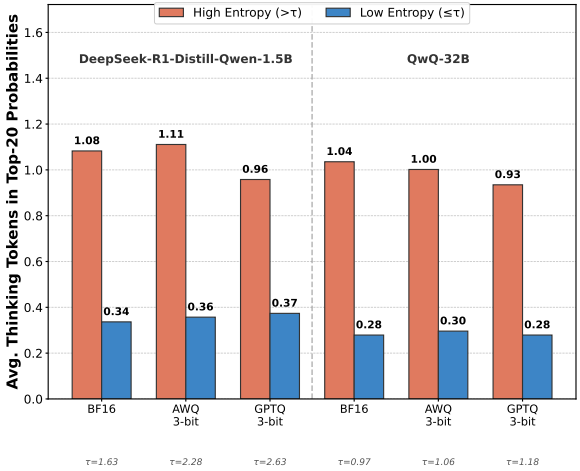
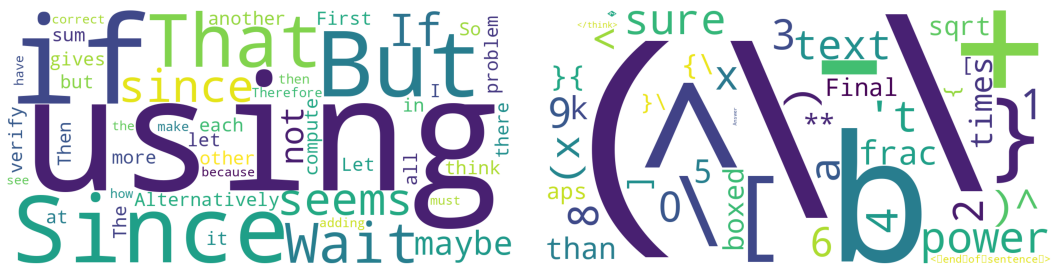


Figure 7: **Thinking tokens are 2–4× more likely to appear among the top-20 next-token predictions at high-entropy positions.** At each decoding step, we measure whether a thinking token from the 50-token list in Table 8 appears in the top-20 most probable next tokens, and compare high-entropy positions (entropy $> \tau$) to low-entropy positions (entropy $\leq \tau$). We set τ to the 80th percentile of the per-model entropy distribution similarly to Wang et al. [39], with τ being reported under each bar. Across DeepSeek-R1-Distill-Qwen-1.5B and QwQ-32B, including their 3-bit quantized variants (AWQ and GPTQ), thinking tokens are substantially more likely to be ranked among the top predictions precisely when the model is locally uncertain. Notably, these top-20 sampling rates of thinking tokens at high-entropy positions are high despite the large vocabulary sizes (151,936 for DeepSeek-R1-Distill-Qwen-1.5B; 152,064 for QwQ-32B), indicating that high-entropy positions concentrate probability mass onto tokens that can trigger overthinking. Experiments are on MATH-500.

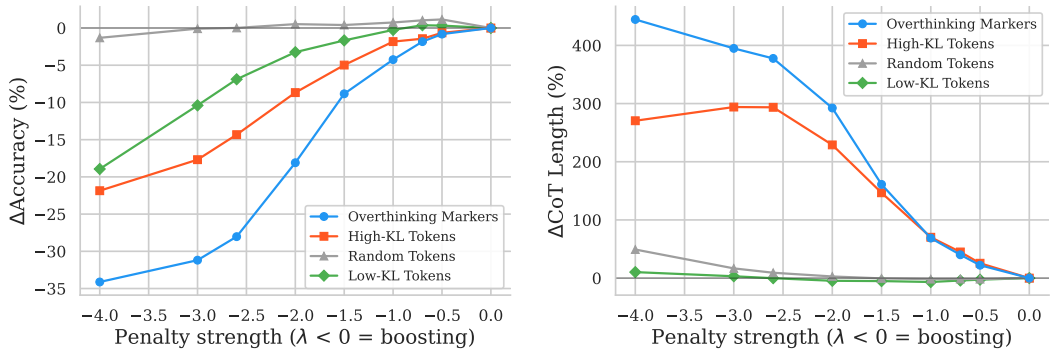


(a) High-KL tokens (overthinking markers) (b) Low-KL tokens (math tokens)

Figure 8: **Word clouds of the 50 highest-KL and 50 lowest-KL tokens between BF16 and AWQ 3-bit on MATH-500.** Token size reflects average KL divergence. High-KL tokens express thinking and branching. Low-KL tokens represent mathematical content.

B.4 Negative Penalty Sweep: Boosting Overthinking Markers

To verify that the direction of the penalty matters, we run an additional sweep with negative λ values that boost rather than suppress the target tokens. Figure 9 shows accuracy and CoT length changes when boosting each token list on Qwen-1.5B, averaged across all six quantization configurations and our five benchmarks. Boosting overthinking markers and high-KL tokens causes severe degradation in both accuracy and CoT length: at $\lambda = -3.0$, overthinking markers drop accuracy by 31% while increasing CoT by 395%, and high-KL tokens show a similar effect with an 18% accuracy drop and 294% CoT increase. The similarity between these two lists further confirms that high-KL tokens are a good proxy for the tokens that drive overthinking under quantization. Random tokens remain nearly neutral across all negative λ values, consistent with the positive λ results. Low-KL tokens reduce accuracy at strong boost but barely increase CoT length, reaching only +10% even at $\lambda = -4.0$. Combined with the positive λ results in Figure 5, these results confirm that overthinking markers and high-KL tokens are the ones that control CoT length: suppressing them reduces CoT, boosting them increases it.



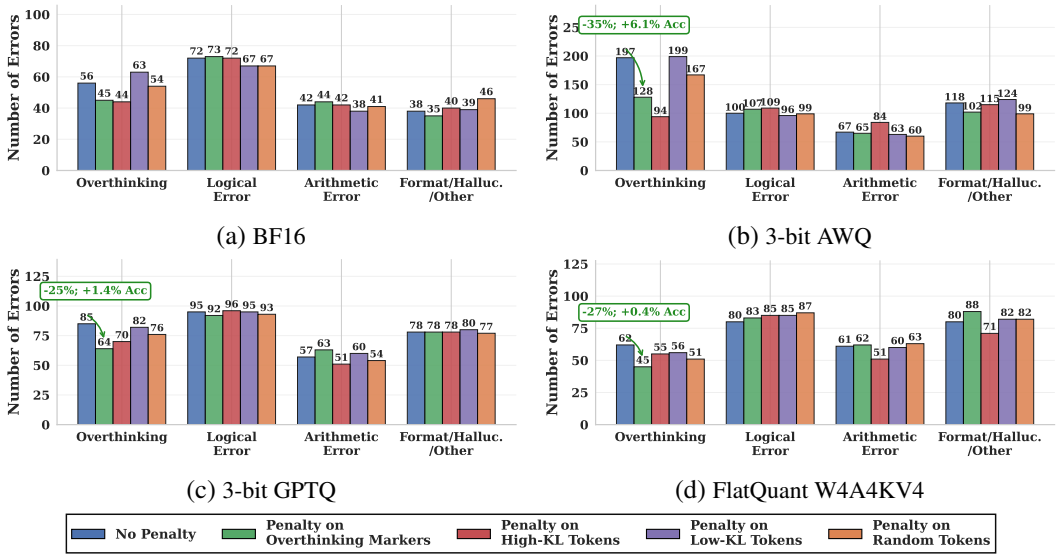
(a) Accuracy when boosting tokens (b) CoT length when boosting tokens

Figure 9: **Boosting overthinking markers increases CoT length and degrades accuracy.** Negative λ boosts rather than suppresses the target tokens. Overthinking markers and high-KL tokens cause severe CoT increases (up to +445%) and accuracy drops (up to −34%). Random and low-KL tokens have minimal effect on CoT length. Averaged across six quantization configurations and five benchmarks on Qwen-1.5B.

B.5 Error Analysis on GSM8K

We repeat the error analysis from Section 4.2 on GSM8K (1,319 questions) to verify that the overthinking pattern generalizes beyond MATH-500. Figure 10 shows the error breakdown for BF16 and three extreme quantization configurations. The same pattern holds: quantization increases overthinking errors, and penalizing overthinking markers reduces them. On AWQ 3-bit, overthinking

errors drop from 197 to 128 (−35%) while accuracy improves by 6.1%. As on MATH-500, penalizing random or low-KL tokens does not consistently reduce overthinking errors and can actually increase them.



(a) BF16 (b) 3-bit AWQ

(c) 3-bit GPTQ (d) FlatQuant W4A4KV4

Figure 10: **Error analysis on GSM8K confirms the overthinking pattern.** Same setup as Figure 3 but on GSM8K. Penalizing overthinking markers reduces overthinking errors by up to 35% on AWQ 3-bit while improving accuracy by 6.1%.

B.6 Full Quantization Evaluation

We report a comprehensive evaluation of PTQ across model scales, datasets, and quantization methods in Table 2. For each base model, we compare the BF16 checkpoint against multiple weight-only quantization settings (GPTQ, AWQ) at 4-bit and 3-bit precision, as well as FlatQuant configurations that quantize weights, activations, and the KV cache. We summarize results using two metrics: task accuracy (Acc) and average chain-of-thought length in tokens (Tok).

Across all models and benchmarks, we observe the same pattern: **more aggressive quantization generally decreases accuracy while increasing CoT length**. This effect is especially pronounced at 3-bit precision, where models often exhibit large CoT increases while simultaneously losing substantial accuracy. In contrast, 4-bit quantization is typically less disruptive: several W4A16KV16 configurations remain close to BF16 accuracy with modest CoT increases. FlatQuant at W8A8KV8 is the most stable setting overall, often matching BF16 accuracy with small CoT changes, whereas W4A4KV4 can reintroduce the same pattern of longer reasoning with degraded accuracy.

B.7 Effect of the Overthinking Penalty

We report the effect of the overthinking penalty for each model individually in Tables 3 to 7. For each quantization configuration, we show baseline accuracy and CoT length (Base row) and the results with the best penalty λ (+Pen row). The Δ columns report the mean change across the five benchmarks with standard deviation across all λ values in the sweep.

The penalty consistently reduces CoT length across all models and quantization configurations. The CoT reduction is largest for the most aggressively quantized configurations (3-bit weight-only and W4A4KV4), where overthinking is most severe, and smallest for mild quantization (4-bit weight-only and W8A8KV8). Accuracy is preserved or improved in the majority of configurations. The few cases where accuracy slightly decreases still show substantial CoT reductions, yielding a net improvement in the efficiency-accuracy tradeoff. Larger models (Qwen-14B, QwQ-32B) show smaller absolute CoT reductions but the penalty remains effective, confirming that the intervention generalizes across model scales.

Table 2: **Effect of quantization on accuracy and CoT length across models and benchmarks.** We report accuracy (%) and average CoT length (tokens) for BF16 and each quantization configuration. More aggressive quantization generally decreases accuracy while increasing CoT length.

Model	Quantization	MATH-500		GSM8K		AIME-90		LiveCodeBench		GPQA-Diamond		Avg.		Δ vs BF16	
		Acc	Tok	Acc	Tok	Acc	Tok	Acc	Tok	Acc	Tok	Acc	Tok	Acc	Tok
Qwen 1.5B	BF16	85.6	5156	84.2	2664	24.4	17465	17.5	19066	30.8	11887	48.5	11248	0.0	0
	GPTQ W4A16KV16	82.8	6192	82.4	3979	17.8	17770	12.7	20529	30.3	13202	45.2	12334	-3.3	+1087
	GPTQ W3A16KV16	71.6	6813	76.1	3996	8.9	20458	6.3	25854	26.3	18010	37.8	15026	-10.7	+3779
	AWQ W4A16KV16	83.0	6321	83.5	3732	17.8	18123	12.7	23711	34.3	13422	46.3	13062	-2.2	+1814
	AWQ W3A16KV16	47.0	23444	63.5	16263	4.4	33015	4.5	38793	25.8	28726	29.0	28048	-19.5	+16801
	FlatQuant W8A8KV8	84.8	5008	84.3	2719	22.2	17727	17.2	19947	36.9	12083	49.1	11497	+0.6	+249
	FlatQuant W4A4KV4	66.4	8041	78.5	3409	7.8	17515	7.5	25383	30.8	20813	38.2	15032	-10.3	+3785
Qwen 7B	BF16	93.2	3937	91.1	1882	45.6	12733	37.7	16478	50.0	9911	63.5	8988	0.0	0
	GPTQ W4A16KV16	92.6	4584	91.1	2233	45.6	14306	38.4	16048	58.1	10903	65.2	9615	+1.6	+627
	GPTQ W3A16KV16	91.0	4831	90.1	2525	30.0	16969	27.2	19139	41.9	11417	56.0	10976	-7.5	+1988
	AWQ W4A16KV16	93.4	3977	90.6	1959	43.3	14359	36.2	15787	47.5	10636	62.2	9344	-1.3	+355
	AWQ W3A16KV16	91.2	4842	90.4	2142	30.0	16148	26.9	21288	48.0	10656	57.3	11015	-6.2	+2027
	FlatQuant W8A8KV8	93.0	3840	90.4	1919	40.0	13412	35.4	15526	49.5	9880	61.7	8915	-1.9	-73
	FlatQuant W4A4KV4	83.6	5444	87.7	2120	23.3	20409	12.3	23862	50.0	17957	51.4	13958	-12.1	+4970
Qwen 14B	BF16	94.6	3735	94.1	1734	57.8	13116	49.6	14435	57.6	9489	70.7	8502	0.0	0
	GPTQ W4A16KV16	95.6	3437	93.3	1641	62.2	11193	52.6	15313	60.1	9387	72.8	8194	+2.0	-308
	GPTQ W3A16KV16	94.4	3607	92.8	1592	44.4	12397	42.2	14643	53.5	9130	65.5	8274	-5.3	-228
	AWQ W4A16KV16	95.0	3626	92.6	1646	56.7	12426	48.1	14751	59.6	9340	70.4	8358	-0.3	-144
	AWQ W3A16KV16	93.0	4277	93.0	2064	48.9	14331	42.2	17533	56.6	10458	66.7	9733	-4.0	+1231
	FlatQuant W8A8KV8	94.8	3826	94.2	1728	61.1	11895	51.9	14008	57.6	9402	71.9	8172	+1.2	-330
	FlatQuant W4A4KV4	95.0	3675	93.5	1802	52.2	12667	49.3	15221	63.6	8752	70.7	8423	-0.0	-78
Llama 8B	BF16	89.0	4629	89.0	2192	44.4	14533	35.1	16176	49.0	10225	61.3	9551	0.0	0
	GPTQ W4A16KV16	89.6	4640	88.6	2327	41.1	14123	35.1	15467	48.0	10727	60.5	9457	-0.8	-94
	GPTQ W3A16KV16	82.2	6181	85.4	2362	26.7	18230	24.6	24081	29.8	12819	49.7	12735	-11.6	+3184
	AWQ W4A16KV16	89.2	4763	88.0	2206	30.0	15470	35.1	16893	48.0	10458	58.1	9958	-3.2	+407
	AWQ W3A16KV16	80.4	5357	82.6	2428	12.2	14787	23.1	18136	35.9	9631	46.8	10068	-14.5	+517
	FlatQuant W8A8KV8	88.8	4595	89.6	2144	37.8	14410	37.3	15641	48.5	10611	60.4	9480	-0.9	-71
	FlatQuant W4A4KV4	86.4	4988	87.6	2355	22.2	15405	35.1	17833	40.9	10239	54.4	10164	-6.9	+613
QwQ 32B	BF16	97.2	4494	95.7	1913	76.7	15345	62.3	17914	63.1	10071	79.0	9947	0.0	0
	GPTQ W4A16KV16	97.4	4450	95.6	1855	78.9	15710	58.6	19034	59.6	9914	78.0	10193	-1.0	+245
	GPTQ W3A16KV16	96.8	4902	95.4	1973	64.4	16073	50.4	19090	56.1	9252	72.6	10258	-6.4	+311
	AWQ W4A16KV16	97.4	4486	95.8	1936	75.6	15733	59.0	19111	62.6	10767	78.1	10407	-0.9	+459
	AWQ W3A16KV16	96.0	4612	95.9	1815	60.0	16687	51.5	19493	61.1	10121	72.9	10546	-6.1	+598

Table 3: **Effect of the overthinking penalty on Qwen-1.5B.** For each quantization configuration, the top row (Base) shows accuracy (%) and CoT length (thousands of tokens) without penalty, and the bottom row (+Pen) shows results with the best penalty λ. Per-benchmark cells are colored when the penalty changes accuracy or CoT length by $\geq$2%. ΔAcc and ΔLen% report the mean change across the five benchmarks with std across λ values.

			AIME	MATH	GSM8K	GPQA	LCB	Avg.	ΔAcc	ΔLen%
			Acc (%) / Len (k)					*Acc / Len*	*mean*$_{\pm std}$	
Qwen-1.5B	**BF16**	Base	**24.4 / 17.5**	**85.6 / 5.2**	**84.2 / 2.7**	**30.8 / 11.9**	**17.5 / 19.1**	**48.5 / 11.2**		
		+Pen	25.6 / 14.8	85.8 / 4.5	85.1 / 2.6	39.4 / 10.5	17.5 / 18.3	50.7 / 10.1	$+2.1_{\pm 1.6}$	$-9.2_{\pm 5.7}$
	GPTQ W4	Base	17.8 / 17.8	82.8 / 6.2	82.4 / 4.0	30.3 / 13.2	12.7 / 20.5	45.2 / 12.3		
		+Pen	24.4 / 15.3	84.8 / 4.8	83.2 / 2.8	34.8 / 10.7	14.6 / 16.4	48.4 / 10.0	$+3.2_{\pm 1.2}$	$-20.9_{\pm 7.2}$
	GPTQ W3	Base	8.9 / 20.5	71.6 / 6.8	76.1 / 4.0	26.3 / 18.0	6.3 / 25.9	37.8 / 15.0		
		+Pen	11.1 / 15.6	73.0 / 5.8	77.5 / 2.8	28.8 / 12.5	8.2 / 22.2	39.7 / 11.8	$+1.9_{\pm 1.1}$	$-22.6_{\pm 6.2}$
	AWQ W4	Base	17.8 / 18.1	83.0 / 6.3	83.5 / 3.7	34.3 / 13.4	12.7 / 23.7	46.3 / 13.1		
		+Pen	23.3 / 16.7	84.0 / 4.9	84.2 / 3.2	38.4 / 12.0	14.9 / 19.4	49.0 / 11.2	$+2.7_{\pm 1.5}$	$-14.8_{\pm 5.2}$
	AWQ W3	Base	4.4 / 33.0	47.0 / 23.4	63.5 / 16.3	25.8 / 28.7	4.5 / 38.8	29.0 / 28.0		
		+Pen	8.9 / 29.3	61.2 / 12.9	69.5 / 7.6	30.3 / 22.7	6.3 / 35.3	35.2 / 21.6	$+6.2_{\pm 1.7}$	$-28.0_{\pm 7.5}$
	FQ W8A8	Base	22.2 / 17.7	84.8 / 5.0	84.3 / 2.7	36.9 / 12.1	17.2 / 19.9	49.1 / 11.5		
		+Pen	26.7 / 15.3	85.4 / 4.6	85.1 / 2.2	38.9 / 10.6	15.7 / 18.4	50.4 / 10.2	$+1.3_{\pm 1.5}$	$-11.8_{\pm 4.5}$
	FQ W4A4	Base	7.8 / 17.5	66.4 / 8.0	78.5 / 3.4	30.8 / 20.8	7.5 / 25.4	38.2 / 15.0		
		+Pen	12.2 / 16.8	71.4 / 6.2	78.9 / 2.1	31.3 / 17.6	8.6 / 23.8	40.5 / 13.3	$+2.3_{\pm 1.3}$	$-17.6_{\pm 4.9}$

Table 4: **Effect of the overthinking penalty on Qwen-7B.** For each quantization configuration, the top row (Base) shows accuracy (%) and CoT length (thousands of tokens) without penalty, and the bottom row (+Pen) shows results with the best penalty λ. Per-benchmark cells are colored when the penalty changes accuracy or CoT length by $\geq$2%. ΔAcc and ΔLen% report the mean change across the five benchmarks with std across λ values.

			AIME	MATH	GSM8K	GPQA	LCB	Avg.	ΔAcc	ΔLen%
			Acc (%) / Len (k)					*Acc / Len*	*mean*$_{\pm std}$	
Qwen-7B	**BF16**	Base	**45.6 / 12.7**	**93.2 / 3.9**	**91.1 / 1.9**	**50.0 / 9.9**	**37.7 / 16.5**	**63.5 / 9.0**		
		+Pen	52.2 / 11.2	94.2 / 3.5	91.6 / 1.6	56.6 / 9.4	38.8 / 14.9	66.7 / 8.1	$+3.2_{\pm 1.9}$	$-10.0_{\pm 2.7}$
	GPTQ W4	Base	45.6 / 14.3	92.6 / 4.6	91.1 / 2.2	58.1 / 10.9	38.4 / 16.0	65.2 / 9.6		
		+Pen	46.7 / 13.3	93.4 / 3.8	91.5 / 1.9	51.5 / 10.6	35.8 / 15.4	63.8 / 9.0	$-1.4_{\pm 1.7}$	$-9.3_{\pm 2.8}$
	GPTQ W3	Base	30.0 / 17.0	91.0 / 4.8	90.1 / 2.5	41.9 / 11.4	27.2 / 19.1	56.0 / 11.0		
		+Pen	34.4 / 14.8	91.2 / 4.2	90.8 / 1.8	48.5 / 10.3	26.5 / 19.2	58.3 / 10.1	$+2.2_{\pm 1.5}$	$-13.1_{\pm 3.0}$
	AWQ W4	Base	43.3 / 14.4	93.4 / 4.0	90.6 / 2.0	47.5 / 10.6	36.2 / 15.8	62.2 / 9.3		
		+Pen	46.7 / 13.0	93.8 / 3.9	91.2 / 1.9	50.5 / 9.9	34.7 / 15.2	63.4 / 8.8	$+1.2_{\pm 1.7}$	$-5.0_{\pm 2.9}$
	AWQ W3	Base	30.0 / 16.1	91.2 / 4.8	90.4 / 2.1	48.0 / 10.7	26.9 / 21.3	57.3 / 11.0		
		+Pen	28.9 / 14.6	90.2 / 4.0	90.5 / 1.8	49.0 / 10.1	28.7 / 18.9	57.5 / 9.9	$+0.2_{\pm 1.4}$	$-12.0_{\pm 3.9}$
	FQ W8A8	Base	40.0 / 13.4	93.0 / 3.8	90.4 / 1.9	49.5 / 9.9	35.4 / 15.5	61.7 / 8.9		
		+Pen	47.8 / 12.3	93.8 / 3.6	91.2 / 1.6	53.0 / 9.9	38.1 / 14.6	64.8 / 8.4	$+3.1_{\pm 2.4}$	$-7.8_{\pm 2.5}$
	FQ W4A4	Base	23.3 / 20.4	83.6 / 5.4	87.7 / 2.1	50.0 / 18.0	12.3 / 23.9	51.4 / 14.0		
		+Pen	22.2 / 19.7	86.4 / 4.7	88.5 / 2.1	51.5 / 14.9	14.2 / 23.2	52.6 / 12.9	$+1.2_{\pm 2.4}$	$-7.7_{\pm 4.8}$

Table 5: **Effect of the overthinking penalty on Qwen-14B.** For each quantization configuration, the top row (Base) shows accuracy (%) and CoT length (thousands of tokens) without penalty, and the bottom row (+Pen) shows results with the best penalty λ. Per-benchmark cells are colored when the penalty changes accuracy or CoT length by $\geq$2%. ΔAcc and ΔLen% report the mean change across the five benchmarks with std across λ values.

			AIME	MATH	GSM8K	GPQA	LCB	Avg.	ΔAcc	ΔLen%
			Acc (%) / Len (k)					*Acc / Len*	*mean*$_{\pm std}$	
Qwen-14B	**BF16**	Base	**57.8 / 13.1**	**94.6 / 3.7**	**94.1 / 1.7**	**57.6 / 9.5**	**49.6 / 14.4**	**70.7 / 8.5**		
		+Pen	63.3 / 10.7	95.6 / 3.5	94.5 / 1.7	62.6 / 8.8	54.5 / 13.4	74.1 / 7.6	$+3.4_{\pm 1.7}$	$-8.5_{\pm 2.3}$
	GPTQ W4	Base	62.2 / 11.2	95.6 / 3.4	93.3 / 1.6	60.1 / 9.4	52.6 / 15.3	72.8 / 8.2		
		+Pen	56.7 / 11.3	95.8 / 3.1	94.2 / 1.5	65.2 / 9.5	51.1 / 14.5	72.6 / 8.0	$-0.2_{\pm 1.8}$	$-4.7_{\pm 2.7}$
	GPTQ W3	Base	44.4 / 12.4	94.4 / 3.6	92.8 / 1.6	53.5 / 9.1	42.2 / 14.6	65.5 / 8.3		
		+Pen	47.8 / 11.9	93.8 / 3.4	92.5 / 1.5	56.1 / 8.7	47.4 / 14.6	67.5 / 8.0	$+2.0_{\pm 1.7}$	$-4.8_{\pm 2.3}$
	AWQ W4	Base	56.7 / 12.4	95.0 / 3.6	92.6 / 1.6	59.6 / 9.3	48.1 / 14.8	70.4 / 8.4		
		+Pen	56.7 / 11.8	96.0 / 3.3	94.2 / 1.5	62.6 / 9.5	52.2 / 13.7	72.3 / 8.0	$+1.9_{\pm 1.6}$	$-5.9_{\pm 2.7}$
	AWQ W3	Base	48.9 / 14.3	93.0 / 4.3	93.0 / 2.1	56.6 / 10.5	42.2 / 17.5	66.7 / 9.7		
		+Pen	47.8 / 14.1	94.2 / 3.9	92.9 / 1.7	56.1 / 10.2	45.9 / 16.2	67.4 / 9.2	$+0.6_{\pm 1.7}$	$-7.5_{\pm 3.5}$
	FQ W8A8	Base	61.1 / 11.9	94.8 / 3.8	94.2 / 1.7	57.6 / 9.4	51.9 / 14.0	71.9 / 8.2		
		+Pen	62.2 / 11.2	95.8 / 3.6	94.2 / 1.6	64.6 / 8.8	51.9 / 13.0	73.7 / 7.6	$+1.8_{\pm 2.1}$	$-7.1_{\pm 2.9}$
	FQ W4A4	Base	52.2 / 12.7	95.0 / 3.7	93.5 / 1.8	63.6 / 8.8	49.3 / 15.2	70.7 / 8.4		
		+Pen	53.3 / 12.6	94.6 / 3.6	93.9 / 1.6	61.6 / 8.8	49.3 / 14.3	70.5 / 8.2	$-0.2_{\pm 1.3}$	$-4.5_{\pm 3.5}$

Table 6: **Effect of the overthinking penalty on Llama-8B.** For each quantization configuration, the top row (Base) shows accuracy (%) and CoT length (thousands of tokens) without penalty, and the bottom row (+Pen) shows results with the best penalty λ. Per-benchmark cells are colored when the penalty changes accuracy or CoT length by $\geq$2%. ΔAcc and ΔLen% report the mean change across the five benchmarks with std across λ values.

			AIME	MATH	GSM8K	GPQA	LCB	Avg.	ΔAcc	ΔLen%
			Acc (%) / Len (k)					*Acc / Len*	*mean*$_{\pm std}$	
Llama-8B	**BF16**	Base	**44.4 / 14.5**	**89.0 / 4.6**	**89.0 / 2.2**	**49.0 / 10.2**	**35.1 / 16.2**	**61.3 / 9.6**		
		+Pen	42.2 / 13.8	91.0 / 4.2	88.6 / 2.0	50.5 / 10.1	38.4 / 15.3	62.1 / 9.1	$+0.8_{\pm 2.0}$	$-5.6_{\pm 2.7}$
	GPTQ W4	Base	41.1 / 14.1	89.6 / 4.6	88.6 / 2.3	48.0 / 10.7	35.1 / 15.5	60.5 / 9.5		
		+Pen	36.7 / 13.8	89.2 / 4.3	88.6 / 2.1	52.0 / 10.2	35.8 / 15.8	60.5 / 9.2	$+0.0_{\pm 2.2}$	$-4.7_{\pm 2.1}$
	GPTQ W3	Base	26.7 / 18.2	82.2 / 6.2	85.4 / 2.4	29.8 / 12.8	24.6 / 24.1	49.7 / 12.7		
		+Pen	28.9 / 16.2	83.4 / 5.6	85.8 / 2.0	39.9 / 12.7	26.9 / 23.0	53.0 / 11.9	$+3.2_{\pm 1.9}$	$-7.9_{\pm 3.9}$
	AWQ W4	Base	30.0 / 15.5	89.2 / 4.8	88.0 / 2.2	48.0 / 10.5	35.1 / 16.9	58.1 / 10.0		
		+Pen	32.2 / 13.2	86.2 / 4.4	88.1 / 1.8	50.5 / 10.4	35.4 / 15.7	58.5 / 9.1	$+0.4_{\pm 1.6}$	$-9.3_{\pm 2.5}$
	AWQ W3	Base	12.2 / 14.8	80.4 / 5.4	82.6 / 2.4	35.9 / 9.6	23.1 / 18.1	46.8 / 10.1		
		+Pen	16.7 / 13.6	77.6 / 4.7	83.7 / 2.0	36.9 / 8.6	23.1 / 15.9	47.6 / 9.0	$+0.7_{\pm 1.5}$	$-12.2_{\pm 2.7}$
	FQ W8A8	Base	37.8 / 14.4	88.8 / 4.6	89.6 / 2.1	48.5 / 10.6	37.3 / 15.6	60.4 / 9.5		
		+Pen	40.0 / 13.9	89.6 / 4.3	89.2 / 2.0	50.0 / 9.8	39.9 / 15.7	61.7 / 9.1	$+1.3_{\pm 1.7}$	$-4.7_{\pm 2.6}$
	FQ W4A4	Base	22.2 / 15.4	86.4 / 5.0	87.6 / 2.4	40.9 / 10.2	35.1 / 17.8	54.4 / 10.2		
		+Pen	22.2 / 14.7	84.2 / 4.9	86.1 / 2.2	48.5 / 9.8	31.7 / 17.3	54.5 / 9.8	$+0.1_{\pm 1.7}$	$-4.1_{\pm 3.0}$

Table 7: **Effect of the overthinking penalty on QwQ-32B.** For each quantization configuration, the top row (Base) shows accuracy (%) and CoT length (thousands of tokens) without penalty, and the bottom row (+Pen) shows results with the best penalty λ. Per-benchmark cells are colored when the penalty changes accuracy or CoT length by $\geq$2%. ΔAcc and ΔLen% report the mean change across the five benchmarks with std across λ values.

			AIME	MATH	GSM8K	GPQA	LCB	Avg.	ΔAcc	ΔLen%
			Acc (%) / Len (k)					*Acc / Len*	*mean*$_{\pm std}$	
QwQ-32B	**BF16**	Base	**76.7 / 15.3**	**97.2 / 4.5**	**95.7 / 1.9**	**63.1 / 10.1**	**62.3 / 17.9**	**79.0 / 9.9**		
		+Pen	81.1 / 12.3	98.0 / 4.0	96.0 / 1.8	66.7 / 9.5	61.9 / 17.3	80.7 / 9.0	$+1.7_{\pm 1.2}$	$-9.6_{\pm 2.6}$
	GPTQ W4	Base	78.9 / 15.7	97.4 / 4.5	95.6 / 1.9	59.6 / 9.9	58.6 / 19.0	78.0 / 10.2		
		+Pen	76.7 / 13.9	97.8 / 4.2	96.1 / 1.7	64.1 / 9.4	61.2 / 17.7	79.2 / 9.4	$+1.2_{\pm 1.1}$	$-7.7_{\pm 2.7}$
	GPTQ W3	Base	64.4 / 16.1	96.8 / 4.9	95.4 / 2.0	56.1 / 9.3	50.4 / 19.1	72.6 / 10.3		
		+Pen	65.6 / 15.0	96.4 / 4.3	95.9 / 1.8	60.1 / 9.5	51.9 / 18.0	74.0 / 9.7	$+1.3_{\pm 1.4}$	$-5.9_{\pm 3.9}$
	AWQ W4	Base	75.6 / 15.7	97.4 / 4.5	95.8 / 1.9	62.6 / 10.8	59.0 / 19.1	78.1 / 10.4		
		+Pen	80.0 / 14.2	96.8 / 4.2	95.9 / 1.8	65.7 / 11.0	60.8 / 18.8	79.8 / 10.0	$+1.8_{\pm 1.1}$	$-4.8_{\pm 2.8}$
	AWQ W3	Base	60.0 / 16.7	96.0 / 4.6	95.9 / 1.8	61.1 / 10.1	51.5 / 19.5	72.9 / 10.5		
		+Pen	62.2 / 14.5	96.6 / 4.2	95.5 / 1.7	60.6 / 8.7	53.4 / 18.7	73.7 / 9.5	$+0.7_{\pm 1.2}$	$-9.6_{\pm 2.9}$

C Derivation of the Local Perturbation Analysis

We provide the short derivations behind the simplified analysis in Section 6, where $p_t = \mathrm{softmax}(z_t)$, $q_t = \mathrm{softmax}(z_t + \epsilon_t)$, and $\epsilon_t \sim \mathcal{N}(0, \sigma_q^2 I)$.

Expected KL divergence. As a function of ϵ_t, the KL divergence $D_{\mathrm{KL}}(p_t \,\|\, q_t)$ is zero at $\epsilon_t = 0$, its gradient $\mathrm{softmax}(z_t + \epsilon_t) - p_t$ is also zero at $\epsilon_t = 0$, and its Hessian at $\epsilon_t = 0$ is $\mathrm{diag}(p_t) - p_t p_t^\top$. A second-order Taylor expansion therefore gives $D_{\mathrm{KL}}(p_t \,\|\, q_t) \approx \frac{1}{2}\epsilon_t^\top \big(\mathrm{diag}(p_t) - p_t p_t^\top\big)\epsilon_t$. Taking the expectation over ϵ_t gives

$$\mathbb{E}_{\epsilon_t}\big[D_{\mathrm{KL}}(p_t \,\|\, q_t)\big] \approx \frac{\sigma_q^2}{2}\,\mathrm{tr}\big(\mathrm{diag}(p_t) - p_t p_t^\top\big) = \frac{\sigma_q^2}{2}\Big(1 - \sum_v p_t(v)^2\Big).$$

Flip probability. Let v_1 and v_2 be the tokens with the two largest logits. The runner-up overtakes the top token when $\epsilon_t(v_2) - \epsilon_t(v_1) > m_t$. Since $\epsilon_t(v_2) - \epsilon_t(v_1) \sim \mathcal{N}(0, 2\sigma_q^2)$, this happens with probability $\Phi\big(-m_t/(\sqrt{2}\sigma_q)\big)$, which gives the lower bound. Every other token v has a logit gap $z_t(v_1) - z_t(v) \geq m_t$, so it overtakes the top token with probability at most $\Phi\big(-m_t/(\sqrt{2}\sigma_q)\big)$, and a union bound over the $|\mathcal{V}| - 1$ competing tokens gives the upper bound.

D Token Lists Used in Experiments

We report the four token lists (50 tokens each) used in our experiments. The overthinking markers list is used for the main penalty intervention. The other three lists are used as controls in the ablation study (Section 5.2). Tokens prefixed with "_" include a leading space in the tokenizer, making them distinct token IDs from their space-free variants.

Overthinking markers (50 tokens). Lexical items that semantically express hesitation, redirection, or local backtracking. The list is constructed based on our inspection of tokens frequently sampled at high-KL decoding positions and is consistent with prior work that defines a similar class of hesitation tokens [14].

Overthinking markers (50 tokens)
`_perhaps`, `_maybe`, `_wait`, `_Wait`, `_actually`, `_hold`, `_Hmm`, `_hmm`, `_Alternatively`, `_alternatively`, `_However`, `_however`, `_instead`, `_Instead`, `_But`, `_but`, `_though`, `_although`, `_yet`, `_rather`, `_unless`, `_otherwise`, `_nonetheless`, `_nevertheless`, `_regardless`, `_still`, `_anyway`, `_Or`, `_or`, `_either`, `_whether`, `_uncertain`, `_unsure`, `_possibly`, `_might`, `_could`, `_another`, `_different`, `_reconsider`, `_rethink`, `_backtrack`, `_retry`, `_recheck`, `_revisit`, `_doubt`, `_confused`, `_wrong`, `_mistake`, `_error`, `_incorrect`.

Table 8: Overthinking markers used in the penalty intervention. All 50 tokens include a leading space (denoted by "_").

High-KL tokens (50 tokens). The 50 tokens with the highest average KL divergence between BF16 and AWQ 3-bit on MATH-500, filtered to tokens appearing at least 50 times. This list overlaps partially with the overthinking markers but also includes functional tokens such as "_So" and "_The" that are unrelated to overthinking.

High-KL tokens (50 tokens)
`_using`, `_if`, `_Since`, `_But`, `_That`, `_Wait`, `_since`, `_If`, `_seems`, `_maybe`, `_not`, `Alternatively`, `But`, `_The`, `_more`, `Wait`, `So`, `_verify`, `_let`, `_in`, `_So`, `First`, `_problem`, `_each`, `_another`, `_but`, `_\n\n`, `_Then`, `_think`, `_it`, `_all`, `_Let`, `_at`, `_I`, `_there`, `_sum`, `_compute`, `_other`, `_gives`, `_Therefore`, `_then`, `_make`, `_must`, `_see`, `_correct`, `_how`, `_the`, `_adding`, `_because`, `_have`.

Table 9: The 50 tokens with the highest average KL divergence between BF16 and AWQ 3-bit. Tokens with "_" have a leading space.

Low-KL tokens (50 tokens). The 50 tokens with the lowest average KL divergence, filtered to tokens appearing at least 50 times. These are predominantly mathematical symbols, digits, and formatting tokens.

Low-KL tokens (50 tokens)
_Answer, </think>, <EOS>, [\n, {, }\, aps, Final, **\n, sqrt, boxed, b, {\, }{, _than, times,]\n\n, k, 5, frac, (x, text, 6, x, __, 3, 8, }, 2,)^, 9, 0, 1, a, _<, _sure, 't,), _power, 4, }\n, \[, _\, _-, _+, _b, ^, \, (\.

Table 10: The 50 tokens with the lowest average KL divergence. These mathematical and formatting tokens remain stable under quantization.

Random tokens (50 tokens). 50 randomly selected tokens serving as a null control.

Random tokens (50 tokens)
_Course, _Examples, _Entity, _teacher, _");, _-type, _James, _softly, _";, _yat, _:The, _Mend, _Proposed, _.ident, _elda, _(_;,, _preseason, _ID, _sort, _fn, _eng, _Emoji, _accessing, _urniture, _Normally, _REUTERS, _notifying, _ResourceManager, _delayed, _Applying, _supporter, _Hot, _build, _PERTIES, _noticing, _crops, _goalie, _Petr, __commit, _.validators, _.Trace, _cut, _propose, _Porno, _point, _zend, _Call, _(pass, _UserId, _picture.

Table 11: 50 randomly selected tokens used as a null control. All 50 include a leading space ("_").

E GPT-5 Prompt for Error Categorization

We provide the prompt used to categorize incorrect generations into the four error types reported in Figure 3 and Figure 10. The prompt instantiates three fields: (i) the original problem, (ii) the model-generated reasoning trace and final answer, and (iii) the gold answer. The judge is instructed to assign *exactly one* label among the four categories and to select *overthinking* whenever the gold answer appears in the intermediate reasoning but is not produced as the final answer. Otherwise, the model has to categorize the error in the other 3 categories. Among all experiments conducted using GPT-5 as judge, the model never failed to choose exactly one category and return it. The judge achieves >95% agreement with human annotations on a subset of the incorrect questions.

F Compute Resources

All inference experiments are run on NVIDIA H100 and H200 GPUs with tensor parallelism across 4 GPUs per job. We use vLLM for inference with default decoding (T=0.6, top-p=0.95).

Quantization. GPTQ and AWQ quantization use the GPTQModel and AutoAWQ libraries, respectively. FlatQuant quantization uses the authors' released code. Quantizing a single model takes approximately 1–4 hours on 4 GPUs depending on model size.

Penalty sweep (full model evaluation). The full penalty sweep across 5 models, 6 quantization configurations (plus BF16 baselines), 9 λ values, and 5 benchmarks totals approximately 1,040 inference jobs. Each job runs on 4 GPUs. Estimated wall-clock times per job range from approximately 1 hour for 1.5B–8B models to 4 hours for QwQ-32B.

Token list ablation. The ablation comparing 4 token lists across 6 quantization configurations, 5 benchmarks, and 8 λ values on Qwen-1.5B totals 960 inference jobs plus 30 baselines, each on 4 GPUs.

Negative penalty sweep. The negative λ sweep mirrors the ablation grid (960 jobs on Qwen-1.5B).

Prompt for GPT-5 Judge

You are an expert at analyzing mathematical reasoning traces from a given model to identify error types.

Classify the error into exactly ONE of these categories:

(A) overthinking: The model reaches the correct solution at some point but does NOT commit to it. It instead opens new reasoning paths or excessively questions the assumptions that it made.

(B) logical_error: The model follows an incorrect plan or misunderstands the problem from early on.

(C) arithmetic_error: The overall approach/plan is correct, but the model makes concrete computational/-calculation mistakes.

(D) formatting_hallucination_other: The model produces an off-format answer, introduces hallucinated constraints, or exhibits other behavioral failures not covered above.

Problem:
problem

Model's Reasoning and Final Answer:
model_output

Correct (Gold) Answer:
gold_answer

If at any point in the chain-of-thought, the model finds the correct (gold) answer but does not commit to it as the final answer, choose overthinking (A). Otherwise, analyze the reasoning and determine if the error is a logical error (B), an arithmetic error (C), or a formatting/hallucination/other error (D). Respond with your final answer as a single letter: A, B, C, or D.

Table 12: Prompt template used for GPT-5 error categorization.

KL divergence analysis. Computing the position-level KL divergence between BF16 and AWQ 3-bit requires running both models in prefill mode on all MATH-500 generations on 4 H100 GPUs.

Error analysis. The GPT-5 error categorization runs via OpenAI API. Each categorization call processes one incorrect answer at negligible compute cost compared to inference.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We provide empirical evidence in the main text and appendix that support all the claims made in the abstract and introduction, and reflect the same numbers reported in them.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations in the conclusion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Section 6 states the assumptions of our local perturbation analysis, namely isotropic Gaussian noise on the logits and a second-order approximation of the KL divergence, and Section C provides the short derivations of the expected KL approximation and of the flip-probability bounds.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide all the generation hyperparameters, calibration datasets, judge prompt, overthinking marker list, and all other details needed to reproduce our results in the main text and appendix.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The datasets used in this work are publicly available, and the implementations of the quantization methods we evaluate are also publicly available. Our simple logit penalty is easy to implement with the details we provided. **We plan to release the code in the near future.**

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We believe that we provide all the details needed to reproduce our results and understand our experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report the standard deviation on both accuracy and chain of thought length improvements across penalties in our tables.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report GPU types (NVIDIA H100/H200), tensor parallelism configuration (4 GPUs per job) and software frameworks used. See Section F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We reviewed the code of ethics and believe we do follow it in this work.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: Our work improves the efficiency of quantized reasoning models, reducing inference cost and making deployment more accessible.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: We do not release new models or datasets. Our work analyzes existing publicly available models and proposes an inference-time intervention that requires no model modification.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the original papers for all models, quantization methods, benchmarks, and calibration datasets. All models and datasets used are publicly available under their respective open-source licenses.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not release new datasets, models, or code as part of this submission. The overthinking markers token list is documented in Section D.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We use GPT-5 as an LLM judge to classify the type of errors that quantized models make. We provide the full prompt used in Section E.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.