# When Quantization Preserves Generation but Degrades Verification: A Frozen-Candidate Study Across Model Scale

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

## Abstract

Quantized language models are commonly evaluated as generators, yet the same models are increasingly used to judge, rerank, and select candidate answers. These roles need not respond identically to reduced precision. We study this distinction with a frozen-candidate protocol: for each model, eight candidates are generated once in FP16 and then held byte-identical while the verifier is evaluated at FP16 and several quantized precisions. Correctness is assigned by deterministic semantic checking, so verifier comparisons are not confounded by a changing candidate pool. On 300 MATH-500 problems, Qwen2.5-7B-Instruct provides a clear example. HQQ 8-bit changes greedy generation accuracy from $0.467$ to $0.473$ ($\Delta = +0.006$), while global verifier AUROC falls from $0.825$ to $0.544$ and within-problem ranking from $0.698$ to $0.411$. Best-of-eight selection falls from $0.500$ to $0.430$, below the exact random-selection rate of $0.455$. The same 7B degradation is observed with bitsandbytes int8 and NF4. In contrast, the HQQ 8-bit result does not show a comparable verification drop at Qwen2.5-3B, -14B, or -32B; the 3B result is additionally sensitive to the quantizer, with NF4 producing a separate degradation. A Llama-3.1-8B experiment is not a strong cross-family test because its FP16 verifier is already near chance ($\mathrm{AUROC} = 0.471$). Output-level analysis of the 7B model shows an approximately $8\times$ reduction in correct-versus-incorrect margin separation and a higher decision-flip rate among correct candidates than incorrect candidates. The evidence therefore supports a narrower claim than a universal "quantization breaks verification" hypothesis: generation accuracy alone does not certify preservation of verifier ranking, and verification should be evaluated directly for the model, precision, interface, and candidate set used downstream.

## 1 Introduction

A language model can be asked to do two different things with the same mathematical problem. It can *solve* the problem, producing an answer, or it can *judge* a proposed solution and decide whether the answer is correct. Modern inference pipelines increasingly combine the two: generate several candidates, score them with a verifier, and keep the candidate receiving the highest score ( Cobbe et al., 2021; Lightman et al., 2024). Quantization is attractive in exactly these settings because lower precision reduces memory and inference cost, and a substantial literature has developed around post-training and weight-only quantization of large language models (Yao et al., 2022; Xiao et al., 2023; Frantar et al., 2022; Lin et al., 2023; Egiazarian et al., 2024; Shao et al., 2024). The usual evaluation question is consequently straightforward: does the quantized model retain its generation accuracy?

That question is necessary, but it is not sufficient for a verifier-dependent system. Suppose a model continues to solve roughly the same fraction of problems after quantization, but its scores no longer separate correct candidates from incorrect ones. A generation benchmark would report almost no change, while a best-of-$N$ pipeline could silently stop working. Conversely, if verification remains intact, a generation-only degradation at a very low bit width tells a different story. The two capabilities therefore need to be measured separately.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

This paper asks whether such a dissociation can occur. We deliberately make the question falsifiable. We do not assume that verification must be more fragile than generation, nor do we assume that an effect found in one checkpoint generalizes to its siblings. Instead, we vary precision while holding the candidate material fixed, then repeat the experiment across model sizes and quantizers.

The central design choice is simple. For problem $i$, we first generate a pool $\mathcal{C}_i = \{x_{i1}, \ldots, x_{iK}\}$ at FP16. We then freeze this pool and ask the same model, loaded at different precisions, to judge exactly the same candidates. If the quantized model gives different scores, that difference is a change in the *verifier*, not a consequence of giving it different answers to inspect. This separation is the main methodological contribution of the study.

The resulting story is not monotonic. In Qwen2.5-7B-Instruct, 8-bit quantization shows little detectable generation change while strongly degrading verification. The effect survives three independently implemented quantizers. But the same experiment does not reproduce the cliff at Qwen2.5-3B, -14B, or -32B. Thus the data support a model-specific failure mode, not a universal property of low-bit inference. This distinction matters because a single positive result is otherwise easy to over-generalize.

Contributions.

- We introduce an auditable frozen-candidate protocol that separates generation from verification while using deterministic external correctness labels.
- We document a large 8-bit verification degradation in Qwen2.5-7B-Instruct while detecting no comparable degradation in greedy generation.
- We test the effect across three sibling Qwen2.5 sizes, two additional quantizers, and a second model family, showing where the effect does and does not appear.
- We analyze the 7B verifier's output margins and decision flips to characterize the observed ranking collapse.

## 2 Related Work

*Mathematical verification.* Verifier-based evaluation asks a model to distinguish correct from incorrect solutions rather than relying only on whether one generated answer is correct. Cobbe et al. ( Cobbe et al., 2021) trained verifiers for mathematical word problems, and Lightman et al. (Lightman et al., 2024) demonstrated the value of step-level verification for mathematical reasoning. These studies motivate treating verification as a capability in its own right, while recent reward-model benchmarks likewise emphasize that evaluator quality must be measured directly rather than inferred from the generator (Lambert et al., 2025). Our question is narrower: holding candidate solutions fixed, does inference-time quantization alter a model's ability to rank them?

*Quantization and evaluator reliability.* Quantization methods such as HQQ (Badri and Shaji, 2024) and bitsandbytes/QLoRA-style quantization (Dettmers et al., 2023) reduce memory and computation, but their effects are commonly summarized using generation quality or task accuracy. Separately, work on evaluator reliability emphasizes that an evaluator's scoring behavior is a distinct object of study (Zheng et al., 2023; Liu et al., 2023; Dutta et al., 2024; Murugadoss et al., 2025; Li et al., 2024; Chen et al., 2024). Position and presentation biases can also alter LLM judgments, motivating controlled evaluator interfaces (Shi et al., 2025; Chen et al., 2024). We combine these perspectives by measuring generation and candidate discrimination under the same inference-time precision change.

*Positioning.* We do not claim that frozen-candidate evaluation is the first way to study LLM judges, nor that the 7B result is a universal property of quantization. The contribution is an experimentally controlled comparison in which candidate text and external correctness labels are fixed while verifier precision varies. This isolates a verifier-side change from a change in the candidate distribution.

## 3 Problem Formulation

The notation below is intentionally minimal. Each quantity is introduced in plain language before it is used in a formula.

**Candidate labels and verifier score.** We index the $n$ evaluation problems by $i = 1, \ldots, n$ and the $K$ candidates for each problem by $k = 1, \ldots, K$. For problem $i$, let $x_{ik}$ denote candidate $k$. An external checker assigns $y_{ik} = 1$ when the candidate's extracted final answer is equivalent to the gold answer, and $y_{ik} = 0$ otherwise. Thus, $y_{ik}$ is a binary reference label used to test the verifier; it is not produced by the language model. In this study, $n = 300$ and $K = 8$.

For a verifier running at precision setting $b$ (where $b$ identifies the model's precision or quantization arm, such as FP16 or HQQ-8), let $z_Y^{(b)}(x_{ik})$ be the next-token logit assigned to "Yes" and $z_N^{(b)}(x_{ik})$ the logit assigned to "No" for candidate $x_{ik}$. We use their difference as the verifier score:

$$m_{ik}^{(b)} = z_Y^{(b)}(x_{ik}) - z_N^{(b)}(x_{ik}). \tag{1}$$

Here $m_{ik}^{(b)}$ is the signed Yes–No logit margin for candidate $k$ of problem $i$. A positive $m$ means the model prefers "Yes," while a larger $m$ means a stronger preference for "Yes." This single number is the score used to rank candidates.

We also convert the two logits into a probability over these two answers. Here $q_{ik}^{(b)}$ denotes the normalized probability assigned to "Yes" when only the "Yes" and "No" verbalizers are considered:

$$q_{ik}^{(b)} = \frac{\exp(z_Y^{(b)})}{\exp(z_Y^{(b)}) + \exp(z_N^{(b)})}. \tag{2}$$

The probability is used for calibration diagnostics; the margin $m$ is used for ranking.

**Global ranking: AUROC.** A verifier is useful when correct candidates tend to receive higher scores than incorrect candidates. AUROC expresses this idea as the probability that a randomly chosen correct candidate receives a higher score than a randomly chosen incorrect candidate. Here $x^+$ denotes a randomly chosen correct candidate, $x^-$ denotes a randomly chosen incorrect candidate, $\Pr(\cdot)$ denotes probability over such candidate pairs, and $m^{(b)}(x)$ is shorthand for the margin of candidate $x$ at precision setting $b$:

$$\mathrm{AUROC}_b = \Pr(m^{(b)}(x^+) > m^{(b)}(x^-)) + \frac{1}{2}\Pr(m^{(b)}(x^+) = m^{(b)}(x^-)). \tag{3}$$

The second term gives half credit for a tie. An AUROC of $0.5$ corresponds to random ranking, while values closer to $1$ indicate better separation.

**Within-problem ranking.** Global AUROC can be affected by differences between problems. We therefore ask a more direct question: *for the same problem, does the verifier put its correct candidates above its incorrect candidates?*

For problem $i$, let $P_i = \{x_{ik} : y_{ik} = 1\}$ be the set of correct candidates and $N_i = \{x_{ik} : y_{ik} = 0\}$ the set of incorrect candidates. We compare every correct–incorrect pair. Let $\mathcal{M}$ denote the set of problems containing at least one candidate of each type; $|\mathcal{M}|$ is therefore the number of mixed-label problems:

$$W_i^{(b)} = \frac{1}{|P_i||N_i|} \sum_{x^+ \in P_i} \sum_{x^- \in N_i} \mathbb{1}[m^{(b)}(x^+) > m^{(b)}(x^-)]. \tag{4}$$

Here $\mathbb{1}[\cdot]$ equals $1$ when the condition inside the brackets is true and $0$ otherwise. Thus $W_i^{(b)}$ is the fraction of correct–incorrect pairs for problem $i$ that are ordered correctly. We average this value over the mixed-label problem set $\mathcal{M}$:

$$W_b = \frac{1}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} W_i^{(b)}. \tag{5}$$

Thus $W_b = 1$ means every correct–incorrect comparison is ordered correctly on average, and $W_b = 0.5$ means the ordering is no better than random. The subscript $b$ is retained because the ranking can change with verifier precision.

**What happens when the verifier must choose one answer?** In practice, a verifier is often used to select the highest-scoring candidate. For each problem $i$, we therefore choose the candidate index $\hat{k}_i^{(b)}$ with the largest verifier margin, where $\arg\max$ returns the index attaining the maximum:

$$\hat{k}_i^{(b)} = \arg\max_k m_{ik}^{(b)}, \qquad A_b^{\mathrm{sel}} = \frac{1}{n}\sum_{i=1}^{n} y_{i\hat{k}_i^{(b)}}. \tag{6}$$

In words, we take the candidate with the largest verifier score and check whether that candidate is correct. $A_b^{\mathrm{sel}}$ is the fraction of the $n$ problems for which this selected candidate is correct.

We also report two simple reference points. If one of the $K$ candidates were chosen uniformly at random, the exact expected accuracy would be

$$A^{\mathrm{rand}} = \frac{1}{n}\sum_i \frac{1}{K}\sum_k y_{ik}. \tag{7}$$

The oracle accuracy, denoted $A^{\mathrm{oracle}}$, is the fraction of the $n$ problems for which at least one of the $K$ candidates is correct. The indicator is 1 for such a problem and 0 otherwise:

$$A^{\mathrm{oracle}} = \frac{1}{n}\sum_i \mathbb{1}\left[\sum_k y_{ik} > 0\right]. \tag{8}$$

We normalize selection between these two reference points. The denominator is the maximum improvement available from the random baseline to the oracle for this fixed candidate pool:

$$S_K(b) = \frac{A_b^{\mathrm{sel}} - A^{\mathrm{rand}}}{A^{\mathrm{oracle}} - A^{\mathrm{rand}}}. \tag{9}$$

The interpretation is simple: $S_K(b) = 0$ means the verifier performs exactly like random selection, while $S_K(b) = 1$ would match the best possible choice from the generated pool. Negative values mean the verifier's selected candidate is correct less often than the exact random baseline.

**Definition of the dissociation.** The experiment looks for one specific pattern: generation changes very little while verifier ranking changes substantially. Let $G_b$ denote greedy generation accuracy at precision setting $b$, and let $G_{\mathrm{FP16}}$ and $W_{\mathrm{FP16}}$ denote the corresponding FP16 generation accuracy and within-problem ranking. We summarize the paired changes as

$$\Delta G_b = G_b - G_{\mathrm{FP16}}, \qquad \Delta W_b = W_b - W_{\mathrm{FP16}}. \tag{10}$$

Here $\Delta G_b$ is the change in generation accuracy and $\Delta W_b$ is the change in within-problem verifier ranking relative to FP16. A small $|\Delta G_b|$ together with a large negative $\Delta W_b$ is the empirical signature we call a verification dissociation. We use "cliff" only when this drop is large relative to the model's own FP16 verifier.

## 4 Experimental Design

### 4.1 Frozen candidates and correctness

We use $n = 300$ problems from MATH-500 (Hendrycks et al., 2021). Throughout the experiment, $G$ denotes greedy generation accuracy, i.e., the fraction of these $n$ problems for which the greedily decoded answer is correct. MATH is a standard benchmark for evaluating mathematical problem solving, while verifier studies have separately emphasized the value of evaluating candidate correctness rather than generation alone ( Cobbe et al., 2021; Lightman et al., 2024; Wang et al., 2023). For each model, eight candidates are sampled once at FP16 with temperature $0.8$, top-$p = 0.95$, and at most 512 new tokens. Every candidate is stored with its problem identifier, text, generation settings, SHA-256 hash, and checker output. The resulting 2,400 candidates are then reused at every precision.

Correctness is determined by a deterministic semantic checker with LaTeX/expression extraction rather than string matching. A candidate whose extracted final answer is equivalent to the gold answer is labeled correct. Extraction failures and checker errors are retained and treated consistently

Table 1: Candidate-label accounting for the corrected 7B frozen pool. The categories are mutually exclusive in the stored records.

| Category | Count |
|---|---|
| Total candidate records | 2,400 |
| Parsed successfully | 2,168 |
| Parsed and correct | 1,093 |
| Parsed and not equivalent | 1,075 |
| Gold-parse failure category | 232 |

rather than selectively removed. This choice does not claim to validate every intermediate proof step; the target is final-answer correctness, which is the same target used to evaluate generation and verifier discrimination.

The verifier prompt is deliberately minimal. This choice also makes the evaluation closer to a controlled binary judge setting than to rubric-based or free-form evaluator protocols (Zheng et al., 2023; Li et al., 2024). "Problem: $\{problem\}$. Proposed Solution: $\{solution\}$. Is this solution correct? Answer with a single word: Yes or No." We read the next-token logits for the exact single-token verbalizers and verify at load time that both verbalizers are single tokens. We retain the raw margins, probabilities, candidate text, prompt hash, and verbalizer IDs.

### 4.2 Quantization and models

The main comparison uses FP16 and HQQ 8/4/3/2-bit (Badri and Shaji, 2024). We treat these as inference-time compression settings rather than as interchangeable implementations of a single abstract "bit width," because quantizers differ in granularity, calibration, and arithmetic (Frantar et al., 2022; Lin et al., 2023; Xiao et al., 2023; Yao et al., 2022). Only attention/MLP linear projections are replaced; embeddings, normalization layers, and the output head remain in full precision. HQQ uses group size 64 and FP16 weight computation. Independently, we evaluate bitsandbytes int8 and NF4 (Dettmers et al., 2022; 2023). Replacement counts are checked against the expected number of attention/MLP projections, and a finite-logit sanity check is performed before evaluation.

The primary model family is Qwen2.5-Instruct at 3B, 7B, 14B, and 32B (Qwen Team, 2025). Qwen2.5-7B is the model in which the initial pilot suggested a verification cliff, so it receives the most detailed analysis. We additionally test Llama-3.1-8B-Instruct (Dubey et al., 2024) to separate a Qwen-family pattern from a property of a second training lineage.

### 4.3 Uncertainty and audit checks

All confidence intervals are obtained by resampling at the problem level. For bootstrap replicate $r = 1, \ldots, B$, where $B$ is the total number of bootstrap replicates, we draw $n = 300$ problems with replacement and retain all $K = 8$ candidates belonging to each sampled problem. For a metric $T$, the empirical interval is the 2.5th and 97.5th percentiles of the bootstrap values $T^{(1)}, \ldots, T^{(B)}$, where $T^{(r)}$ is the value of metric $T$ on bootstrap replicate $r$. Thus $r$ indexes a bootstrap replicate and $T$ denotes whichever reported metric is being resampled. This preserves within-problem dependence and avoids pretending that the 2,400 candidate judgments are independent observations.

The study also includes three audit checks. First, the exact random baseline is computed analytically rather than estimated from a random draw. Second, the same frozen candidate hashes are used across verifier precision arms. Third, the 7B cliff is tested with three quantization implementations. These checks are important because an apparent verifier failure can otherwise be produced by a changed candidate distribution or by a quantizer-specific implementation artifact.

## 5 Results

### 5.1 A clear dissociation in Qwen2.5-7B

The first result is the motivating case. At FP16, Qwen2.5-7B has generation accuracy $G = 0.467$, global verifier AUROC $0.825$, and within-problem ranking $W = 0.698$. At HQQ 8-bit, generation

| FP16 generation: $K = 8$ candidates per problem | | |
|---|---|---|
| Frozen candidate pool: 2,400 candidate records | | |
| *FP16 verifier* | *HQQ 8-bit verifier* | *bnb int8 / NF4 verifier* |

Figure 1: Controlled evaluation layout. Candidates are generated once at FP16 and frozen. Each verifier arm receives the same candidate texts and external correctness labels; only verifier precision and quantization implementation vary.

Table 2: Qwen2.5-7B-Instruct on 300 MATH-500 problems. $W$ is within-problem ranking over 84 mixed-label problems. $S_8$ uses the exact random baseline $0.455$. Intervals are problem-bootstrap 95% CIs where available.

| Arm | $G$ | AUROC | $W$ | $A^{\text{sel}}$ | $S_8$ |
|---|---|---|---|---|---|
| FP16 | .467 | .825 | .698 | .500 | .301 |
| HQQ-8 | .473 | .544 | .411 | .430 | -.172 |
| bnb-int8 | .460 | .530 | .399 | .423 | -.217 |
| bnb-NF4 | .470 | .518 | .412 | .443 | -.082 |
| HQQ-4 | .460 | .595 | .455 | .450 | -.059 |
| HQQ-3 | .383 | .490 | .398 | .443 | -.194 |
| HQQ-2 | .017 | .473 | .539 | .470 | .121 |

is $0.473$, while AUROC falls to $0.544$ and $W$ falls to $0.411$. The generation difference is $+0.006$ with a problem-bootstrap 95% interval of $[-0.010, 0.023]$, so we detect no comparable generation degradation in this arm. The within-problem difference is $-0.287$ with interval $[-0.374, -0.205]$. Thus the same fixed model shows little detectable change in generation while losing most of its verifier ranking signal.

The downstream selection statistic tells the same story. FP16 selects a correct candidate on $50.0\%$ of problems. HQQ 8-bit selects correctly on $43.0\%$, below the exact random-selection rate of $45.5\%$. The normalized utility changes from $0.301$ to $-0.172$. This is not a claim that the quantized model is adversarial: its AUROC remains slightly above $0.5$. The point is that a verifier that was useful at FP16 has become worse than uniform selection on this particular candidate pool: negative $S_8$ means that the selected candidate performs below the exact random-selection baseline.

The effect is not confined to HQQ. Bitsandbytes int8 gives AUROC $0.530$ and $W = 0.399$, while NF4 gives AUROC $0.518$ and $W = 0.412$. Their greedy generation accuracies are $0.460$ and $0.470$, respectively. All three quantizers therefore show no comparably large generation degradation while producing a substantial verification loss at 7B. This makes a single-library implementation artifact an incomplete explanation for the 7B result.

At 4-bit, the separation remains visible: generation changes by $-0.007$ while $W$ changes by $-0.243$ with bootstrap interval $[-0.328, -0.161]$. At 3- and 2-bit, however, generation itself deteriorates severely, so these arms cannot be interpreted as clean tests of verification preservation. This distinction is important: the paper's strongest evidence comes from a regime where generation is essentially unchanged.

### 5.2 The cliff does not scale across Qwen2.5

The next question is whether the 7B result is a generic consequence of reducing precision. We repeat the same protocol on Qwen2.5-3B, -14B, and -32B. The answer is no. Table 3 compares FP16 and HQQ 8-bit. Generation changes by at most $0.003$ in magnitude, but only 7B shows a large verification change.

For 3B, AUROC moves from $0.670$ to $0.684$ and $W$ from $0.565$ to $0.571$. For 14B, AUROC is unchanged at $0.772$ and $W$ rises from $0.597$ to $0.604$. For 32B, AUROC changes from $0.562$ to $0.563$ and $W$ from $0.385$ to $0.388$. Their bootstrap intervals overlap substantially between FP16 and 8-bit on generation, AUROC, and $W$.

The quantizer cross-check reinforces the model-specific interpretation. At 7B, HQQ-8, bnb-int8, and bnb-NF4 produce AUROCs $0.544$, $0.530$, and $0.518$. At 14B they produce $0.772$, $0.768$, and

Table 3: FP16 versus HQQ 8-bit across Qwen2.5 sizes. Here the subscripts 16 and 8 denote the FP16 and HQQ 8-bit verifier arms, respectively. All $\Delta$ values denote 8-bit minus FP16. The 7B row is the only row with a large verification change in this comparison.

| Model | $G_{16}$ | $G_8$ | $\Delta G$ | $\text{AUROC}_{16}$ | $\text{AUROC}_8$ | $\Delta$AUROC | $W_{16}$ | $W_8$ | $\Delta W$ |
|---|---|---|---|---|---|---|---|---|---|
| 3B | .400 | .403 | +.003 | .670 | .684 | +.014 | .565 | .571 | +.006 |
| 7B | .467 | .473 | +.006 | .825 | .544 | -.281 | .698 | .411 | -.287 |
| 14B | .483 | .480 | -.003 | .772 | .772 | +.000 | .597 | .604 | +.007 |
| 32B | .510 | .510 | +.000 | .562 | .563 | +.001 | .385 | .388 | +.003 |

Table 4: Output-level changes for the 7B FP16/HQQ-8 verifier pair on the common candidate pool.

| Statistic | FP16 | HQQ-8 |
|---|---|---|
| Mean margin, correct candidates | 13.29 | 2.32 |
| Mean margin, incorrect candidates | 3.45 | 1.08 |
| Correct–incorrect separation $D$ | 9.84 | 1.24 |
| ECE | 0.304 | 0.344 |
| Brier score | 0.335 | 0.396 |

$0.777$; at 32B, $0.563$, $0.572$, and $0.624$. The 3B case is instructive: bnb-int8 gives $0.691$, while bnb-NF4 gives $0.452$. This is a smaller, quantizer-sensitive degradation, but it is not the same broad 7B pattern. We therefore do not collapse all low-bit behavior into a single category.

A second observation follows from the FP16 baselines. Verification quality itself is non-monotonic with parameter count: AUROC is $0.670$, $0.825$, $0.772$, and $0.562$ for 3B, 7B, 14B, and 32B. The largest model is the weakest self-verifier in this experiment. This is not a quantization effect, but it warns against using parameter count alone as a proxy for verifier quality.

### 5.3 Output-level analysis of the 7B verifier

The frozen candidates let us inspect the raw signal behind the cliff. For each of the 2,400 common candidates, we compare the FP16 and HQQ-8 margins. The mean separation between correct and incorrect candidates is given by the following equation. Here $\mathbb{E}[\cdot \mid y = c]$ denotes the average verifier margin over candidates whose external correctness label is $c \in \{0, 1\}$, and $D_b$ measures the separation between the two class means:

$$D_b = \mathbb{E}[m^{(b)} \mid y = 1] - \mathbb{E}[m^{(b)} \mid y = 0]. \tag{11}$$

At FP16, the correct-candidate mean margin is $13.29$ and the incorrect-candidate mean is $3.45$, giving $D_{16} = 9.84$. At HQQ-8, the corresponding means are $2.32$ and $1.08$, giving $D_8 = 1.24$. The separation therefore shrinks by approximately a factor of eight.

The change is also asymmetric after thresholding. Here a candidate's binary decision is obtained by comparing its Yes and No logits: "Yes" corresponds to $m_{ik}^{(b)} > 0$ and "No" to $m_{ik}^{(b)} < 0$; an exact zero is a tie. Across all candidates, $28.5\%$ change their binary Yes/No decision between FP16 and HQQ-8. Among truly correct candidates, $45.2\%$ flip; among incorrect candidates, only $14.5\%$ flip. The quantized verifier is consequently not simply adding symmetric noise; it substantially reduces the positive margin assigned to candidates that are externally labeled correct.

The continuous ranking signal is also substantially reordered. Spearman correlation between FP16 and HQQ-8 margins is $\rho = 0.34$ (Pearson $r = 0.51$). A simple monotone rescaling of FP16 scores would preserve ranking and leave $\rho$ near one; the observed value instead indicates that quantization changes the relative ordering of candidates.

Calibration is a secondary diagnostic rather than the primary target. Standard calibration work treats confidence as a separate property from discrimination (Guo et al., 2017). Calibration tells a consistent but secondary story. FP16 has ECE $0.304$ and Brier score $0.335$; HQQ-8 has ECE $0.344$ and Brier score $0.396$. Neither arm is well calibrated as a probability estimator. The main result is therefore not "quantization destroys confidence calibration"; it is the much more direct loss of ranking information used by best-of-$N$ selection.

### 5.4 A SECOND FAMILY: LLAMA-3.1-8B

We finally run the seven-arm protocol on Llama-3.1-8B-Instruct. At FP16, its verifier AUROC is $0.471$ with 95% CI $[0.445, 0.499]$, and HQQ-8 gives $0.472$ with $[0.447, 0.500]$. Generation is $0.340$ at both precisions. The absence of an 8-bit cliff here should not be treated as evidence that the 7B phenomenon cannot occur in other families: this model begins with essentially no detectable verifier discrimination on this benchmark. There is little signal available to lose.

At HQQ-2, the Llama model's greedy generation accuracy is only $0.003$, while its verifier AUROC on the *frozen FP16 candidate pool* rises to $0.674$. Because the candidate pool is fixed, this rise is not caused by a low-bit-generated candidate distribution. It is nevertheless not a test of the paper's central dissociation: generation accuracy has collapsed to $0.003$, so this arm lies outside the generation-preserved regime. We therefore exclude the extreme low-bit Llama arms from claims about preservation of generation alongside verifier change.

## 6 DISCUSSION

The empirical story has three parts. First, the 7B model provides a strong counterexample to the assumption that preserved generation accuracy certifies preserved verification. Second, the effect survives three quantization implementations, which makes a single software implementation a poor explanation. Third, the effect does not transfer to the other Qwen2.5 sizes tested. The correct conclusion is therefore neither "quantization breaks verification" nor "quantization is harmless." It is that verification is a separate capability whose behavior under compression is model-dependent.

This distinction can be expressed as a simple deployment principle. A generation benchmark estimates something like $\Pr[\text{generated answer is correct}]$. A verifier-dependent system also needs $\Pr[m(x^+) > m(x^-)]$ for candidates that the system will compare. These are different functionals of the model. Quantization can perturb the second while leaving the first nearly fixed. No algebraic relationship forces the two changes to have the same magnitude or even the same sign.

The frozen-candidate design is what makes this distinction measurable. Without it, an 8-bit verifier would be judged on an 8-bit-generated candidate pool. If both generation and verification changed, the observed selection failure would have two possible causes. Freezing the candidates removes that ambiguity for the verification comparison. It also makes the best-of-eight statistic easy to interpret: every precision arm is choosing from exactly the same eight alternatives.

The 7B margin analysis suggests a concrete observable failure pattern. The quantized verifier does not merely shift its decision threshold. Its correct-vs-incorrect margin separation falls from $9.84$ to $1.24$, and its per-candidate ranking changes substantially. The asymmetric flip rates then explain why selection suffers: a correct candidate that was strongly favored by FP16 can lose its positive score under quantization, while many incorrect candidates retain their relative status. This is an output-level characterization; the experiment does not identify an internal layer-level causal mechanism.

The non-replication is equally important. The 3B, 14B, and 32B models preserve their verifier metrics at 8-bit even though their generation accuracy is also essentially unchanged. Thus the phenomenon cannot be described as an inevitable consequence of 8-bit quantization, and it cannot be inferred from parameter count. Indeed, full-precision verifier quality is itself non-monotonic across these four sizes.

## 7 LIMITATIONS

The study has several clear boundaries. We use one benchmark, MATH-500, and one candidate-set size, $K = 8$. MATH provides unusually convenient deterministic correctness labels, but open-ended tasks and code verification may behave differently. The verifier interface is also intentionally narrow: a single-token Yes/No decision. Free-form judging, numerical scores, rubrics, or chain-of-thought-based verification may expose different sensitivities.

The quantization is mixed precision rather than end-to-end quantization: embeddings, normalization layers, and the output head remain at full precision. The mechanism analysis is performed only for the 7B FP16 versus HQQ-8 pair. We do not have layer-wise interventions that would establish a

causal internal mechanism. The additional model family contains only one checkpoint and starts at chance-level verification, so it is not a strong test of cross-family generalization.

Finally, each model uses one candidate-pool seed. Problem-level bootstrap intervals quantify uncertainty induced by the sampled evaluation set, not all possible seed-to-seed variation. The study therefore supports a precise claim about the evaluated configurations rather than a universal law about quantized language models.

## 8 Conclusion

We asked whether a quantized language model can preserve generation while losing verification. The answer is yes, but only in the model-specific sense supported by the experiment. Qwen2.5-7B-Instruct shows no detectable degradation in greedy generation at 8-bit while its verifier AUROC and within-problem ranking collapse, and the effect survives HQQ, bitsandbytes int8, and NF4. Raw logits show an $8\times$ reduction in correct-vs-incorrect margin separation and a strong asymmetry in decision flips.

At the same time, the identical protocol finds no comparable 8-bit degradation in Qwen2.5-3B, -14B, or -32B. Llama-3.1-8B starts with a near-chance verifier, making its apparent non-replication uninformative for the main generalization question. The practical lesson is consequently narrow and testable: if a quantized model will be used to verify, rerank, or select model outputs, generation accuracy alone is not an adequate certificate. Verification must be measured directly, on the model, precision, interface, and candidate distribution that the downstream system will actually use.

## Reproducibility Statement

The experiments use a single resumable pipeline across all model arms. The configuration is fixed at 300 MATH-500 problems, eight candidates per problem, sampling temperature $0.8$, top-$p = 0.95$, maximum 512 generated tokens, and a common seed of 20260909. The verifier prompt is rendered through each model's native chat template and truncated to at most 1,600 input tokens. Python 3.12.3, PyTorch 2.7.0a0+ecf3bae40.nv25.02, CUDA 12.8, Transformers 5.16.1, Accelerate 1.14.0, HQQ 0.2.8.post1, bitsandbytes 0.50.2, and datasets 5.0.1 were used on an NVIDIA B200 MIG 3g.90gb partition. Raw candidate text, hashes, checker outputs, verifier logits, arm metadata, and bootstrap summaries were retained. The other four model runs resolve model checkpoints at runtime rather than pinning explicit snapshot hashes, which is a minor reproducibility limitation.

## References


Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems*, 2022.

Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. Extreme compression of large language models via additive quantization. In *International Conference on Machine Learning*, 2024.

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

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

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. RewardBench: Evaluating reward models for language modeling. In *Findings of the Association for Computational Linguistics: NAACL*, 2025.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-Eval: NLG evaluation using GPT-4 with better human alignment. In *Proceedings of EMNLP*, 2023.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Gao Peng, Yu Qiao, and Ping Luo. OmniQuant: Omnidirectionally calibrated quantization for large language models. In *International Conference on Learning Representations*, 2024.

Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. Humans or LLMs as the judge? A study on judgement biases. In *Proceedings of EMNLP*, 2024.

Lin Shi, Chiyu Ma, Wenhua Liang, Xingjian Diao, Weicheng Ma, and Soroush Vosoughi. Judging the judges: A systematic study of position bias in LLM-as-a-judge. In *Proceedings of IJCNLP-AACL*, 2025.

Xuezhi Wang et al. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023.

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*, 2023.

Zhewei Yao et al. ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. In *Advances in Neural Information Processing Systems*, 2022.

Lianmin Zheng et al. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems*, 2023.

Xiaonan Li et al. JudgeBench: A benchmark for evaluating LLM-based judges. arXiv preprint, 2024.

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:2110.14168, 2021.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, 2023.

Abhimanyu Dubey et al. The Llama 3 herd of models. arXiv:2407.21783, 2024.

Abhinav Dutta, Sanjeev Krishnan, Nipun Kwatra, and Ramachandran Ramjee. Accuracy is not all you need. In *Advances in Neural Information Processing Systems*, 2024.

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. In *NeurIPS Datasets and Benchmarks Track*, 2021.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Bhuvanashree Murugadoss, Christian Poelitz, Ian Drosos, Vu Le, Nick McKenna, Carina Suzana Negreanu, Chris Parnin, and Advait Sarkar. Evaluating the evaluator: Measuring LLMs' adherence to task evaluation instructions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

Qwen Team. Qwen2.5 technical report. arXiv:2412.15115, 2025.

Hicham Badri and Appu Shaji. HQQ: Half-quadratic quantization of large machine learning models. 2024.