# The Structural Residual Ceiling

*AI Pre-Decoders for the Surface Code*

Manu Bhardwaj

*ifitsmanu.com / Field Notes*

7 May 2026 (v1.0, Field Notes #4)

**Abstract**

A recent NVIDIA preprint (Chamberland, Olle, Li, Thornton, Baratta, April 2026) reports the first AI-based pre-decoder pipeline that simultaneously improves logical error rate (LER) and end-to-end runtime over state-of-the-art surface-code global decoders, with measured speedups up to 3.42× over uncorrelated PyMatching and 3.54× over correlated PyMatching at code distance $d = 31$, $p = 0.006$, on NVIDIA GB300 in FP8 precision. Buried inside that result is an honest negative finding. When the pre-decoder is paired with correlated matching at distances $d \geq 17$, the LER worsens, and applying the authors' separately-trained noise-learning network to pre-decoder residuals fails to recover the gap. The authors state that "nearly all residual errors that lead to a logical fault [...] form strings of length greater than $(d-1)/2$ and which are parallel to the logical observable of interest." This field note argues the cause is not network capacity but the deterministic homological-equivalence canonicalization used to generate training labels. Three concrete, falsifiable mitigations are outlined: randomized rather than deterministic canonicalization, chain-length-aware loss reweighting, and teacher-student distillation with chain-conditioned negative sampling. Each is testable inside the released codebase (NVIDIA/Ising-Decoding, Apache-2.0). Implications for lattice-surgery deployment and NVFP4 quantization-aware training are discussed. No new experiments are reported. This is a perspective.

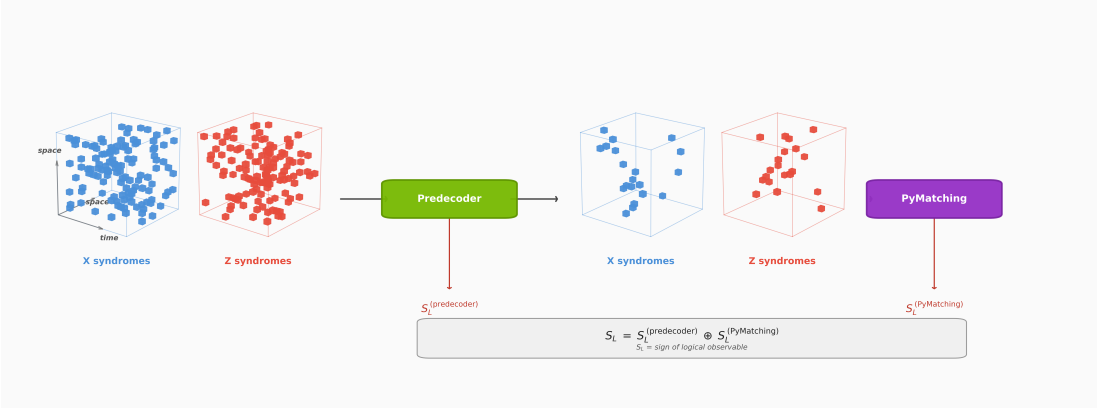


**Figure 1.** Pre-decoder pipeline as illustrated in the NVIDIA/Ising-Decoding repository [2]. The neural network consumes detector syndromes across space and time and predicts corrections that reduce syndrome density before passing residuals to a global matching decoder. Reproduced from the repository under Apache-2.0; not modified.

## 1. Introduction

The case for AI pre-decoders is now empirical, not theoretical. Chamberland et al. [1] report end-to-end runtime savings on GB300 hardware that grow with both code distance and physical error rate, with the largest gains in exactly the regime that matters for early fault-tolerant systems: large $d$, $p$ approaching the surface-code threshold ($\sim$0.7%). The architectural recipe is a fully-convolutional 3D CNN that jointly predicts spacelike (data-qubit) and timelike (measurement) corrections, trained at the receptive-field volume and applied at arbitrary larger volumes via translation equivariance. The released code makes it reproducible [2].

The same paper discloses three limitations the authors do not resolve.

1. The pre-decoder + correlated PyMatching pipeline does not improve LER at $d \geq 17$. A larger residual-network variant ("Model 6", $\sim$42.6 M parameters) reduces the gap but does not close it.
2. The noise-learning network does not improve LER on pre-decoder residuals. This is surprising, because pre-decoding produces non-trivial syndrome statistics that one might expect to benefit from re-learned edge weights.
3. The authors attribute both to a single observation: the residual errors most likely to cause logical faults are long parallel chains, of length exceeding $(d-1)/2$, oriented along the logical observable.

This note argues the observation points to a structural property of the *training labels*, not of the network or the global decoder. Section 2 reviews the relevant background. Section 3 makes the structural-residual claim precise. Section 4 argues that homological-equivalence canonicalization is the proximate cause. Section 5 proposes three mitigations, ordered by implementation cost. Section 6 discusses implications. Section 7 concludes.

All numerical claims are sourced from the original paper or the released code. No new measured quantities are introduced.

## 2. Background

The rotated surface code [4, 5] is a two-dimensional topological quantum error-correcting code whose stabilizers can be measured using nearest-neighbor interactions. Logical operator representatives are weight-$d$ strings; minimum-weight perfect matching (MWPM) decoding [6, 7] succeeds when error chains are shorter than $(d-1)/2$. The circuit-level depolarizing noise model is parameterized by 25 probabilities (4 SPAM, 6 idle channels, 15 CNOT Pauli channels).

A *pre-decoder* is a local operator on the $(d, d, d_m)$ syndrome volume that emits two species of correction. *Spacelike* corrections are Pauli operators applied to data qubits. *Timelike* corrections are bit flips applied to two consecutive rounds of stabilizer measurement outcomes. These corrections rewrite both the syndrome history and the residual data-qubit errors, producing a modified detector configuration that is then handed to a global decoder. In the NVIDIA pipeline, the global decoder is either uncorrelated PyMatching [7] or its correlated two-pass variant [8].

The pre-decoder's training labels are derived by Pauli-frame simulation. At each circuit fault location, the simulator samples a Pauli error, propagates it, and records the resulting space-time correction. Multiple equivalent error chains, chains differing by stabilizer multiplication, produce identical syndrome histories, so the label space is highly degenerate. The

authors apply a homological-equivalence protocol to fix a canonical representative within each class.

The spacelike rule reduces weight-3 X-errors on weight-4 X-stabilizers to weight-1, and maps weight-2 chains deterministically to a fixed side of the stabilizer (e.g. vertical chains rotate to the right column). The timelike rule treats a $Z$ (or $X$) error applied to the same data qubit in two consecutive rounds, plus matching measurement-error flips in the *first* of those two rounds, as a no-op on syndrome history; it is applied iteratively until label sparsity stops decreasing. Both rules are deterministic. Given a sampled error chain, the canonicalization function returns one specific representative every time.

## 3. The structural residual claim, made precise

The empirical claim from the paper is the following. Let $R^{(j)}$ denote the residual data-qubit error left on shot $j$ after applying the pre-decoder's predicted spacelike corrections. Let $F^{(j)} \subseteq R^{(j)}$ be the projection of $R^{(j)}$ onto chains parallel to the logical operator $X_L$ (or $Z_L$) of length $\geq (d-1)/2$. Then:

> *The probability that $R^{(j)}$ causes a logical fault, conditioned on $R^{(j)} \neq \emptyset$, is dominated by the event $F^{(j)} \neq \emptyset$.*

In plain language. Residuals that matter are long, and they are parallel to the logical. Short residuals get cleaned up by the global decoder. Perpendicular long residuals do not flip the logical. Only long parallel ones do.

Two consequences follow.

**MWPM cannot help.** A minimum-weight global decoder, given a long parallel chain, returns a *correct* matching of the residual syndromes. But the correction it applies, having no information about which side of the chain to commit to, flips the logical with probability that approaches 1/2 as chain length increases. The decoder is not making an error. It is making the right call given a degenerate input.

**Noise-learning cannot help either.** Reweighting edges in the matching graph changes which spanning structures the matcher prefers, but does not change whether the matcher *can distinguish* the two homologically equivalent corrections that flip vs. preserve the logical. The information needed to discriminate is, by construction, no longer in the syndrome.

The honest reading is that the pre-decoder is throwing information away at training time. The question is how.

## 4. Diagnosis. Deterministic canonicalization concentrates label mass.

Consider a weight-4 X-stabilizer $g_k(X)$ and a weight-2 X-error $E$ supported on its two rightmost data qubits. By stabilizer multiplication, $E$ is homologically equivalent to $E'$ supported on its two leftmost data qubits (multiply by $g_k(X)$). The canonicalization rule always picks one of these, say $E'$, as the canonical label.

Suppose the underlying physical error was $E$, not $E'$. The training label nevertheless records $E'$. The network learns to predict $E'$ on this syndrome.

For an isolated weight-2 error this is harmless. $E$ and $E'$ produce identical residual syndromes after correction, and either is fine. But the canonicalization compounds across rounds and across chains. A long chain composed of multiple weight-2 segments gets canonicalized segment by segment. Two physically distinct chains, one along the "left" boundary

and one along the "right," can be mapped to the same canonical chain if their stabilizer-weighted decompositions happen to coincide. The network sees the same label for both, has no signal to distinguish them, and at inference, when one of these chains appears with a small perturbation, it produces the canonical correction, which may be the wrong choice for the actual chain.

This is, structurally, the long-parallel-chain failure mode. The chains that fail are exactly the ones for which the canonicalization compounds in the direction of the logical operator, because that is where the most stabilizer-multiplication freedom exists.

This diagnosis is consistent with three pieces of evidence in the paper.

**(a) Adding parameters does not fix the problem.** Model 5 (∼7.1 M) and Model 6 (∼42.6 M) both exhibit the residual structure, with Model 6 reducing but not eliminating the LER gap at $d \geq 17$. If the issue were capacity-limited, scaling would close it.

**(b) The weight-2 timelike extension did not help.** The authors report that the weight-2 timelike homological equivalence extension *did not* improve results during training, and they ship only the weight-1 variant. This is consistent with the hypothesis: extending the canonicalization further concentrates label mass without adding signal.

**(c) Noise-learning helps on raw syndromes but not on residuals.** The noise-learning network *does* improve LER on raw (non-pre-decoded) syndromes, both for correlated and (slightly less) for uncorrelated matching. The same network applied to pre-decoder residuals does not help. The difference is that raw syndromes still carry the structural information the canonicalization erases.

This is an interpretation, not a measurement. It is a hypothesis about the data-generation pipeline that admits direct testing, described next.

## 5. Three mitigations

### *5.1. Randomized canonicalization*

**Cost:** ∼50 lines of change in `code/data/` of the released repo. **Risk:** low.

Replace deterministic `fixEquivalenceX/Z` with a stochastic representative. At training time, when an error $E$ admits multiple homologically equivalent forms, sample one uniformly at random per training shot. The total entropy of the label distribution increases. The network is trained to be invariant under the equivalence class rather than locked to one representative.

Three predictions follow. Per-voxel BCE loss should *increase* (more label noise), but logical-error-rate improvement should *not decrease*, and may improve at large $d$. Long-parallel-chain residuals should decrease in frequency, because the network is no longer biased toward one side of the equivalence class. The improvement should be most pronounced at low $p$, where rare, long chains dominate the residual mix.

If randomized canonicalization improves LER, there is direct evidence that the deterministic choice was the bottleneck. If it does not, the residual structure has another source. This is the cheapest experiment to run and the most informative.

### *5.2. Chain-length-aware loss reweighting*

**Cost:** moderate; requires a chain-detection pass over each label batch. **Risk:** medium.

The voxel-wise BCE loss in the original paper weights every voxel equally. Long parallel chains cause logical faults but contribute the same per-voxel loss as isolated weight-1 errors

that the global decoder cleans up trivially. The training signal is dominated by the easy cases.

Reweight the loss by chain-component length, with a weight $w_\ell \propto \ell$ applied to voxels belonging to a contiguous chain component of length $\ell$. Implementation requires a connected-components pass over each shot's spacelike correction tensor, which is cheap on GPU.

Prediction. Residual long parallel chains decrease, at the cost of increased false-positive corrections on isolated errors. The net LER should improve at distances where logical faults are dominated by long chains ($d \geq 17$ in correlated PM regime).

Risk. Over-weighting can cause the network to "see chains everywhere" and apply spurious corrections. The weight schedule needs tuning, probably as a curriculum (start uniform, ramp up chain weighting).

### *5.3. Distillation with chain-conditioned negative sampling*

**Cost:** highest. Requires a teacher-student pipeline. **Risk:** highest, but with the highest payoff.

The Chamberland paper telegraphs distillation as future work [1]. This note proposes a specific instantiation tailored to the structural residual problem.

The teacher is a deeper, slower network (Model 6 scale or larger) trained on a curriculum that *up-samples* long-parallel-chain failures. Run a baseline Model 5 over a large evaluation set, isolate shots where the global decoder fails, identify the parallel-chain residuals, and oversample those failure modes (and their near-neighbors in syndrome space) in the teacher's training mix.

The student is Model 1 or Model 4 architecture, trained with a hybrid loss combining BCE against ground-truth labels and KL divergence against the teacher's per-voxel sigmoid outputs. The student inherits the teacher's behavior on the rare hard cases without paying the inference cost.

The teacher does not need to be fast. The student does. This is the standard teacher-student decoupling [9] applied to a problem where the long tail, not the typical case, is what limits LER.

Prediction. The student reaches Model 6's LER at $d \leq 13$ with Model 1's runtime. At $d \geq 17$, the student narrows but does not close the gap, because the teacher itself is structurally limited (per Section 4). Combining distillation with randomized canonicalization is the path to closing the gap entirely.

## 6. Implications

### *6.1. Lattice surgery*

The Chamberland paper restricts attention to memory experiments. Lattice surgery [10, 11, 12] introduces merged code patches with effective distances that can exceed 100, and time-boundary detectors that change the input-channel set (the paper notes $N_s > 4$ will be needed).

The structural-residual problem is likely *worse* in lattice surgery, because (i) merged-patch geometries introduce additional homological-equivalence freedom along merge boundaries, and (ii) the relevant logical operators after surgery are products of original logicals, which lengthens the parallel-chain failure mode.

Addressing the structural residual problem is therefore not optional for lattice-surgery deployment. It is on the critical path.

### *6.2. NVFP4 and quantization-aware training*

The paper closes by flagging NVFP4 (4-bit floating point) deployment with quantization-aware training (QAT) as the next runtime frontier. There is a structural concern worth naming. Quantization compounds with label degeneracy. A network trained on degenerate labels has, in some sense, already lost precision before quantization is applied. Further reducing weight precision can amplify the structural-residual failure mode disproportionately.

Concretely, the prediction is that NVFP4 + QAT will degrade LER more on Model 6 + correlated PyMatching than on Model 1 + uncorrelated PyMatching, even though Model 6 has nominally more capacity to absorb quantization noise. The reason is that the loss landscape near long-parallel-chain failures is sharper than near typical errors, and 4-bit precision sees those sharp regions worst.

If this prediction holds, the practical recipe is. Solve the structural-residual problem at FP8 first. Then apply QAT. Not the reverse.

### *6.3. Distance-independence of the structural problem*

The 18 edge-type and 43 hyperedge-type formulas the authors derive are distance-independent. Only the instance counts scale with $d$. The same structural argument applies to the canonicalization rules. Their *form* is fixed; their *coverage* grows with $d$. The structural residual ceiling is therefore not an artifact of the $d \leq 31$ regime tested in the paper. It should be expected to grow with $d$, which is consistent with the LER gap widening at $d \geq 17$.

### *6.4. Marketing numbers reconciled to paper tables*

The NVIDIA Ising solutions page [3] markets the pre-decoder as delivering "2.5x improvement in speed and 3x improvement in accuracy" against state of the art, alongside two pre-trained checkpoints described as "0.9M or 1.8M parameters." These are not the headline numbers from the paper's Tables VIII and X. They are a deliberately conservative subset.

Tracing each marketing claim to the paper:

**Table 1.** Reconciliation of NVIDIA Ising solutions-page marketing numbers to the source rows of [1].

| Marketing claim | Source row | Exact value |
|---|---|---|
| "0.9M / 1.8M parameters" | Table II | Model 1 (912,272), Model 4 (1,797,76 |
| "2.5× speed" | Table VIII, Model 4, $d = 21$, $p = 0.006$ | "2.50x" |
| "3× accuracy" | Table IV, Model 4, $d = 31$, $p = 0.006$ | "3.21x" (LER improvement factor) |

The marketing page quotes one number from the $d = 21$ speedup column and another from the $d = 31$ LER column, both real, both from the same Model 4 checkpoint, but evaluated at different distances. The paper's actual headlines are stronger (3.42× and 4.66× at $d = 31$) but those use Model 5. The marketing chose the smaller shipped checkpoint and reported across two distances. This reconciliation is included so that any reader arriving at the paper from the NVIDIA marketing page is not confused by the apparent discrepancy.

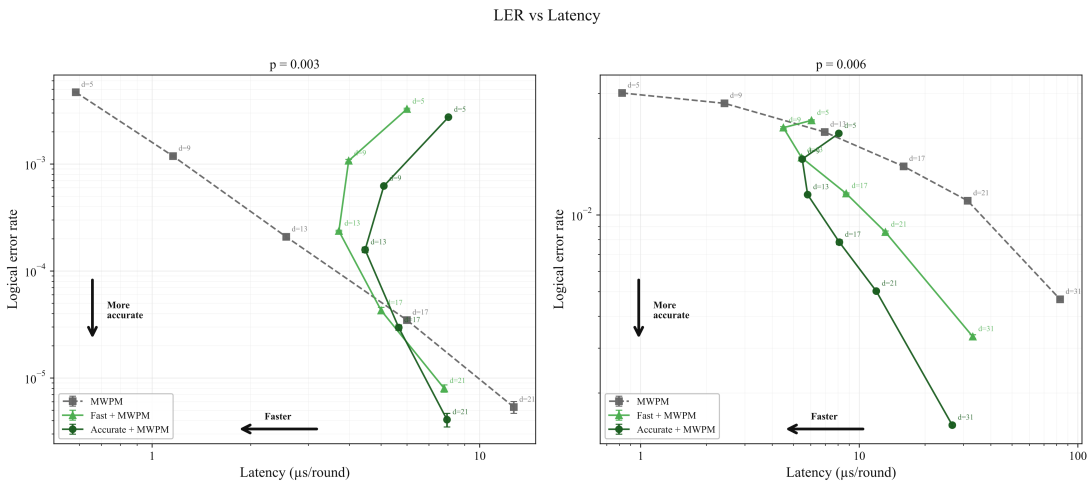


**Figure 2.** End-to-end logical error rate per round vs. single-shot runtime across decoding strategies, at $p = 0.003$ (left) and $p = 0.006$ (right), reproduced from the NVIDIA/Ising-Decoding repository [2]. Each strategy traces a curve over distances $d = 5$ through $d = 31$. The pre-decoder + PyMatching pipeline (M1, M5 with uncorrelated PM; M6 with correlated PM) shifts the achievable frontier downward and to the left at large $d$, where syndrome density dominates the global decoder cost. Points marked with an asterisk in the original figure are LER-extrapolated; runtimes are measured directly. The structural residual ceiling discussed in Section 4 manifests as the M6 curve failing to dominate the corr-PM curve at $d \geq 17$ in the left panel.

## 7. Conclusion

The Chamberland paper is the strongest empirical case yet that AI pre-decoders are practical. Lower LER, lower runtime, on production GPU hardware, with reproducible code. The structural residual ceiling they disclose is a property of the training labels rather than the architecture. Three mitigations are testable inside the released codebase. The cheapest experiment, randomized canonicalization, is also the most informative. Chain-length-aware loss reweighting is the most likely to require curriculum tuning. Distillation is the most expensive but, combined with randomized canonicalization, plausibly closes the LER gap at large $d$.

None of this requires new architectures, larger models, or hardware that does not exist. It requires looking carefully at how training labels are constructed, and noticing that one deterministic choice, the canonicalization side, concentrates label mass in a way the network cannot correct at inference, and that no global decoder downstream can recover.

The broader point is that as decoder pipelines compose multiple learned and algorithmic components, the failure modes that matter migrate from any single component to the *interfaces between them*. The structural residual problem is an interface bug between the pre-decoder's label canonicalization and the global decoder's matching-graph degeneracy. It will not be solved by scaling either component in isolation.

## A. The broader NVIDIA Ising release

The pre-decoder is one component of a larger NVIDIA release branded *Ising* [3]. Two distinct artifacts ship together.

**Ising Decoding** is the subject of this field note: neural pre-decoders for surface codes, available as `Ising-Decoder-SurfaceCode-1-Fast` (R=9, 0.9 M parameters, Model 1 of [1] Table II) and `Ising-Decoder-SurfaceCode-1-Accurate` (R=13, 1.8 M parameters, Model 4 of the same table). Apache-2.0; code at [2].

**Ising Calibration** is a 35 B-parameter (3 B active per token) Mixture-of-Experts vision-language model that analyzes quantum calibration experiment plots and emits structured technical text [13]. Trained in two SFT phases on 72.5 K entries via LLM-augmented data; built on Qwen3.5-35B-A3B; evaluated on the QCalEval benchmark [14] across six question types covering technical description, experimental conclusion, significance, fit-quality assessment, parameter extraction, and experiment-success classification.

The two systems address opposite ends of the quantum-computing AI stack. Ising Decoding accelerates the *real-time* control loop during code execution. Ising Calibration accelerates the *offline* engineering loop that brings a device into specification. They share branding and a release date (April 14, 2026) but no architectural code.

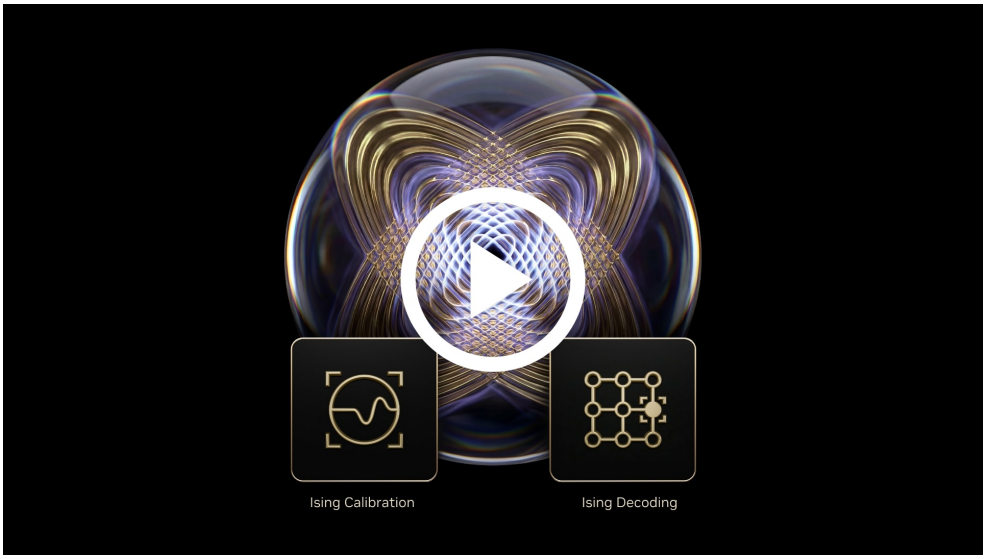


**Figure 3.** NVIDIA's positioning of the Ising release: "Open AI models, training frameworks, data sets, and workflows to the NVIDIA platform for quantum-GPU supercomputing" [3]. This field note concerns only the Decoding component; Figure 4 is included for completeness.

The reason these are not the subject of this field note, despite being part of the same release, is that the structural residual problem analyzed in Sections 3–5 is specific to the geometry of surface-code matching graphs and does not arise in vision-language calibration analysis. The two halves of the Ising release are united by branding and infrastructure, not by shared failure modes. They are mentioned here so that the reader who arrives at NVIDIA's Ising landing page and sees both artifacts knows which one this field note is about, and which one it is not.

## References

[1] Chamberland, C., Olle, J., Li, M., Thornton, S., Baratta, I. (2026). *Fast and accurate AI-based pre-decoders for surface codes.* arXiv:2604.12841. arxiv.org/abs/2604.12841.

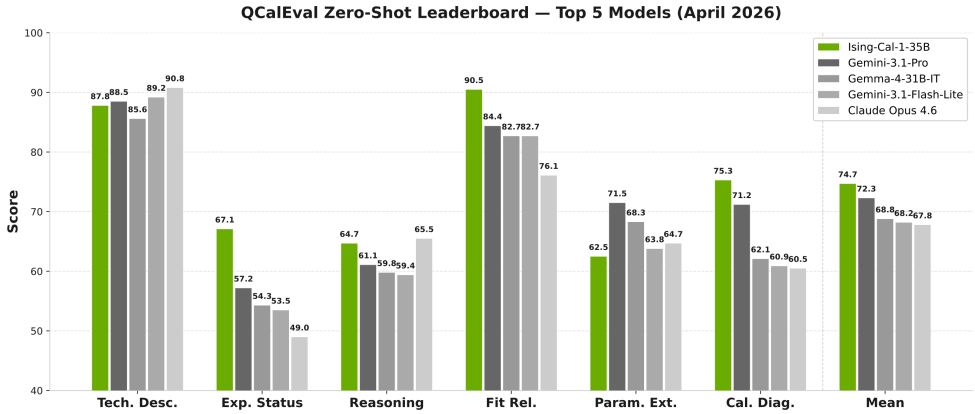


**Figure 4.** Reported QCalEval scores for Ising-Calibration-1-35B-A3B vs. its Qwen3.5-35B-A3B base, reproduced unmodified from the public model card [13]. Six question categories Q1–Q6 are evaluated on 243 entries across 87 scenario types from 22 experiment families (superconducting qubits and neutral atoms), with scores averaged across GPT-5.4 and Gemini-3.1-Pro judges. Headline: 74.7 vs. 55.5 overall.


[2] NVIDIA (2026). *Ising-Decoding: A training framework for AI Quantum Error Correction Decoders.* GitHub repository, Apache-2.0. github.com/NVIDIA/Ising-Decoding.

[3] NVIDIA (2026). *Ising — Open AI Models for Quantum Computing.* nvidia.com/en-us/solutions/quantum-computing/ising.

[4] Dennis, E., Kitaev, A., Landahl, A., Preskill, J. (2002). *Topological quantum memory.* Journal of Mathematical Physics 43, 4452.

[5] Fowler, A. G., Mariantoni, M., Martinis, J. M., Cleland, A. N. (2012). *Surface codes: Towards practical large-scale quantum computation.* Physical Review A 86, 032324.

[6] Edmonds, J. (1965). *Paths, trees, and flowers.* Canadian Journal of Mathematics 17, 449.

[7] Higgott, O. (2022). *PyMatching: a Python package for decoding quantum codes with minimum-weight perfect matching.* ACM Transactions on Quantum Computing 3, 16.

[8] Higgott, O., Gidney, C. (2025). *Sparse Blossom: correcting a million errors per core second with minimum-weight matching.* Quantum 9, 1600.

[9] Hinton, G., Vinyals, O., Dean, J. (2015). *Distilling the knowledge in a neural network.* arXiv:1503.02531. arxiv.org/abs/1503.02531.

[10] Fowler, A. G., Gidney, C. (2018). *Low overhead quantum computation using lattice surgery.* arXiv:1808.06709. arxiv.org/abs/1808.06709.

[11] Litinski, D. (2019). *A Game of Surface Codes: large-scale quantum computing with lattice surgery.* Quantum 3, 128.

[12] Chamberland, C., Campbell, E. T. (2022). *Universal quantum computing with twist-free and temporally encoded lattice surgery.* PRX Quantum 3, 010331.

[13] NVIDIA (2026). *Ising-Calibration-1-35B-A3B* model card. Hugging Face. huggingface.co/nvidia/Ising-Calibration-1-35B-A3B.

[14] Cao, S., Zhang, Z., et al. (2026). *QCalEval: Benchmarking Vision-Language Models for Quantum Calibration Plot Understanding.* NVIDIA Research. research.nvidia.com/publication/2026-04_qcaleval.

---

*Set in Computer Modern 11/15. Typeset with LaTeX. ifitsmanu.com.*