You Cannot Trust a Patch Without a Certificate: Behavioral Certification and Composition Preservation for LLM Capability Repair

Sunil Kumar, Perception. Contact: contact@perception.club. v1 working draft.

> Provenance. Every quantitative claim in this paper is backed by a committed artifact produced against a single SHA-pinned base model. The complete experiment trail — including pre-registered success criteria, refuted hypotheses, and our own incorrect predictions — is the append-only journal kept alongside the code, available on request. Section numbers in brackets (e.g. [J#26]) point to journal entries.

Abstract

Practitioners increasingly *repair* large language models with small, parameter-efficient patches — a LoRA adapter that fixes a specific failure mode, an edit that corrects a behavior — and judge those patches by training loss or a single accuracy number. We argue, and demonstrate, that this is unsafe on two distinct axes. First, a patch that looks excellent by training loss can **leak**: it degrades capabilities it was never meant to touch, in ways a target-metric never reveals. Second, even a good patch is **not preserved under composition**: co-installing it with a second patch can destroy both.

We make the certificate the product. A *behavioral certificate* is a per-dimension PASS/FAIL record — efficacy, locality, regression, and cost — scored exclusively under greedy free generation, with bootstrap confidence intervals, against a base model pinned by commit hash. We pair it with a *composition-preservation test* that re-certifies each patch after co-installation.

On text-to-SQL with Qwen2.5-1.5B- and 7B-Instruct, we report six results. (1) Teacher-forced token agreement overstates free-generation execution accuracy by an amount that is a near-deterministic linear function of the model's free-generation *headroom* — gap = $-0.045 + 0.931 \cdot (1 - \text{free-gen accuracy})$, $R^2 = 0.99$, $r = 0.993$, $p < 10^{-4}$ — and the *same* line holds across a 5× model-size range; an error audit shows the inflation is exactly the family of derailments (spurious joins, hallucinated tables) that teacher forcing masks. The gap is largest precisely in the fallible regime where repair happens and vanishes only at task ceiling, so a certificate must score with free generation. (2) A naive join-repair LoRA whose training loss falls cleanly from 0.55 to 0.165 fails its certificate on three dimensions; 72 of 73 of its collateral failures are a single mechanism, the injection of unwanted joins into queries that need none. (3) The locality threshold cannot be set by fiat; we calibrate it from a benign-perturbation noise floor under a rule fixed before data collection, and show this is both necessary (uncalibrated thresholds reject patches on noise) and sufficient (the calibrated certificate is satisfiable). (4) Naive merging of two individually strong patches drops *both* capabilities below the unpatched base model. (5) This composition conflict is predictable from the patch weights — subspace overlap, delta norms, and an interference ratio — with the counter-intuitive sign that *lower* subspace overlap is *more* destructive; the governing mechanism is destructive interference between patches of comparable magnitude. (6) The optimal install operation is therefore **routing, not merging**: keeping patches separate strictly dominates naive merge across every pair we tested. Finally, on a second, unrelated capability and an external

benchmark — tool/function-call generation on the Berkeley Function-Calling Leaderboard — the same framework refuses a patch that is simultaneously useless on its target (efficacy break-even) and harmful off it (a 0.094 instruction-following regression), and we find the same collateral victim, instruction-following, degraded by *both* SQL and tool-call training.

The certificate framework, the calibration procedure, the conflict analysis, and the full pre-registered experiment record are available on request.

---

1. Introduction

1.1 From "a patch" to "a failure"

The dominant framing of model editing and parameter-efficient fine-tuning begins with a *patch*: a method produces a weight delta, and the delta is evaluated on a benchmark slice. Deployment, however, begins with a *failure*. An operator observes that a model mishandles a class of inputs, classifies the failure, decides whether the underlying capability is even a legitimate repair target, generates a patch, and then must answer the only questions that matter in production: *Did the patch fix the failure without breaking anything else? And will it survive next to the patches already installed?*

These two questions — isolation and composition — are precisely the ones a single benchmark number cannot answer. Our thesis is that the unit of trust in capability repair is not the patch but the **certificate**: a structured, per-dimension, statistically honest statement of what the patch does and does not do, together with a proof that the statement survives composition. The patch is an artifact; the certificate is the product.

1.2 Why the obvious evaluation fails

Two failure modes of naive evaluation motivate the entire paper.

Inflated measurement. Capability is often scored with teacher forcing (feeding the gold continuation and measuring per-token agreement) or with lenient string matching. Both overstate the behavior a user actually experiences, because the user gets free, unguided generation. A patch can look strong under teacher forcing and be weak in deployment. A certificate that inherits this inflation certifies a fiction.

Hidden collateral damage and composition collapse. Even with honest scoring, a patch optimized for one capability can silently move others, and two honest patches can interfere when combined. A target-only metric is blind to both. The literature has begun to notice the first problem — recent work argues editing evaluations dramatically overstate real-world effectiveness and that standard locality metrics are weakly correlated with the strength of the regularizer meant to control them — but no prior framework combines capability- granularity behavioral certificates, composition-preservation testing, and a pre-registered calibration of the very thresholds the verdict depends on.

1.3 Contributions

1. **The behavioral certificate (§3).** A per-dimension PASS/FAIL record — efficacy, locality, regression, patch cost — under a non-negotiable evaluation protocol (greedy free generation, set-semantics

execution accuracy, bootstrap CIs, SHA-pinned base), with dimensions never averaged.

2. **The measuring-stick law (§5).** A pre-registered, audited demonstration that teacher-forced scoring inflates free-generation capability by an amount that is a near-deterministic linear function of the model's headroom (R^2 = 0.99), holding across a 5× model-size range, with the error mechanism identified.
3. **A leaky patch caught and explained (§6).** A LoRA whose training loss looks ideal fails its certificate; we localize 72/73 of its collateral failures to one mechanism.
4. **Threshold calibration as a first-class step (§7).** A rule, fixed before data, that derives the locality threshold from a benign-perturbation noise floor — and the finding that this makes the certificate satisfiable where a hand-set threshold does not.
5. **Composition breaks certificates (§9).** Naive merge of two strong patches drops both below base.
6. **Conflict is weight-predictable, and routing dominates merging (§10–§11).** The conflict tracks subspace overlap (with a reversed sign), delta norms, and interference; and the install operation that recovers the lost capability is routing, with naive merge strictly dominated.
7. **The framework transfers to a second domain (§12).** On tool-call generation and an external benchmark (BFCL), the certificate refuses a patch that is both useless on its target and harmful off it, and surfaces a cross-capability effect — instruction-following degraded by both SQL and tool-call training — that an averaged score would hide.

1.4 Method as a stance

We adopt practices uncommon in empirical deep learning but standard in the experimental sciences. Every threshold and success criterion is **pre-registered** in a dated journal entry before the run that tests it. We record refutations, including a prediction of ours that the data reversed (§10). When a result hinges on a measurement artifact — a probe too small to resolve its own threshold, a floating-point boundary — we say so and fix it *forward only*, never retroactively converting a recorded failure into a pass. The journal is the methods-and-results trail; this paper is its synthesis.

---

2. Related Work

Behavioral vs. parameter-space certification. The closest contemporaneous work, *Provably Safe Model Updates*, certifies updates by constructing invariant regions in parameter space. That guarantee is formal and worst-case but is about parameters; ours is statistical and behavioral, defined directly on greedy generations with confidence intervals at capability granularity. The two are complementary: a parameter-space certificate bounds what *can* change; a behavioral certificate measures what *did* change on the behaviors you declared you care about.

Predicting merge success. *Demystifying Mergeability*, concurrent with this work, predicts model-merging success from interpretable model properties — essentially the question of our §10. We differ in target and framing: our predictand is *certificate degradation* (does a co-install break a PASS dimension?), and our analysis surfaces a specific mechanism — destructive interference between comparable-magnitude deltas — together with the counter-intuitive result that lower subspace overlap is more

harmful. We treat this concurrent work as corroborating, and position our contribution as the certificate-centric and mechanistic account.

Evaluation honesty. *The Mirage of Model Editing* documents that editing evaluations overstate in-the-wild effectiveness, motivating our insistence on free generation (§5). *Are We Evaluating Edit Locality Properly?* shows existing locality/specificity metrics are weakly correlated with regularizer strength and insensitive across methods — independent support for our finding that locality thresholds must be calibrated from a measured noise floor (§7), and a protocol our locality dimension should be benchmarked against.

Editing and merging methods. Locate-and-edit methods (the ROME/MEMIT lineage) target factual associations; weight-space merging methods (Task Arithmetic, TIES, DARE) combine task vectors. We are agnostic to the patch *mechanism* — we evaluate LoRA adapters here — and orthogonal to the merging *method* — we study the naive sum as the baseline an install policy must beat. The exposure-bias gap underlying §5 is classical. Prior work documents that editing can harm general abilities and that routing among parameter-efficient experts trades off against merging; what is new here is the per-dimension certificate that turns those observations into an accept/refuse contract.

---

3. The Certificate Framework

3.1 Capability selection gate

A capability is an eligible repair target only if it is **measurable** (a concrete automatable metric exists), **repeatable** (same input → same score under a fixed seed and greedy decoding), **isolated** (a locality neighborhood of behaviors that should not move can be declared), and **benchmarkable** (a held-out eval slice exists, disjoint from training). Targets failing any criterion — "reasoning," "helpfulness" — are rejected before any patch is trained. This gate prevents benchmark sprawl and is the precondition under which the locality dimension is even meaningful.

3.2 Evaluation protocol (non-negotiable)

All capability scoring uses **greedy free generation**. Teacher forcing is permitted only to *demonstrate* its inflation (§5) and never appears in a certificate path. SQL correctness uses **set-semantics execution accuracy** (a prediction is correct iff it returns the same result set as the gold query, order-insensitive, duplicate-insensitive — the Spider convention). Every reported metric carries its sample size and a bootstrap 95% confidence interval. The base model is pinned by commit hash (`Qwen/Qwen2.5-1.5B-Instruct @ 989aa7980e…`); a certificate evaluated against a different revision is invalid by construction.

3.3 Dimensions and the verdict function

A certificate reports four dimensions, **each adjudicated independently — they are never averaged**:

- **Efficacy.** The change in target-capability execution accuracy, patched minus base, with a paired bootstrap CI. PASS iff the point estimate ≥ τ_eff (default 0.10) **and** the CI lower bound > 0.
- **Locality.** The maximum absolute drift across a *declared-before-running* neighborhood of behaviors that should not move. PASS iff max drift ≤ τ_loc (calibrated; §7). Two-sided: a patch is not permitted

to *move* a neighbor, in either direction, because uncontrolled side effects — even beneficial ones — signal entanglement that breaks composition.

- **Regression.** Per-suite health: a perplexity ratio (≤ 1.03) and held-out general-QA drop (≥ −0.02). Perplexity is teacher-forced by nature but is a health signal, not a capability score, and is sanctioned for regression only.
- **Patch cost.** Trainable parameters, delta L2 norm, storage bytes — recorded, not adjudicated.

Boundary semantics are explicit and consistent: a metric exactly at its threshold PASSES, with a float tolerance so that a quantized count landing on the limit is not rejected by representation error. (This was fixed *forward-only* after we observed a boundary case; §8.4.)

3.4 Composition preservation

Given two certified patches X and Y and an install operator $\oplus$, the composition test re-runs each patch's certificate measurements on the composed model X $\oplus$ Y. The patch is *composition-preserved under* $\oplus$ if no dimension that held individually fails after co-installation. The baseline operator is naive merge (the sum of weight deltas); §11 evaluates alternatives.

---

4. Experimental Setup

Model and data. Qwen2.5-1.5B-Instruct, SHA-pinned, in bf16 (the §5 measuring-stick law also uses the 7B variant). Spider (text-to-SQL): we render each database schema into the prompt and execute predictions against the provided SQLite databases; the second domain (§12) uses the Berkeley Function-Calling Leaderboard. We iterate on 1.5B by design and reserve larger-model runs for recorded results. Capability slices — joins, aggregation, schema-linking — are derived by classifying gold queries; locality also includes non-SQL probes (an MMLU subset; an instruction-following probe) and regression suites (WikiText perplexity; held-out MMLU). Patches are LoRA adapters.

Sampling. Eval slices are seeded *random* subsamples, not prefixes. We discovered mid-project that Spider's dev file is ordered by database, so a first-*n* prefix is a biased database mix (base join accuracy was 0.24 on the first 100 examples versus 0.60 on items 101–150); all post-discovery runs use seeded random sampling, and we flag the one re-baselining boundary this creates. Within-run comparisons (patched vs. base, alone vs. merged on identical examples) are unaffected.

Reproducibility and discipline. Greedy decoding makes every measurement deterministic given the seed; bootstrap CIs use a fixed seed. Certificate runs are resumable via per-pass checkpoints. Each experiment is one runnable entrypoint writing a schema-validated JSON artifact; thresholds and success criteria are pre-registered before the run.

---

5. The Measuring Stick: Free Generation vs. Teacher Forcing

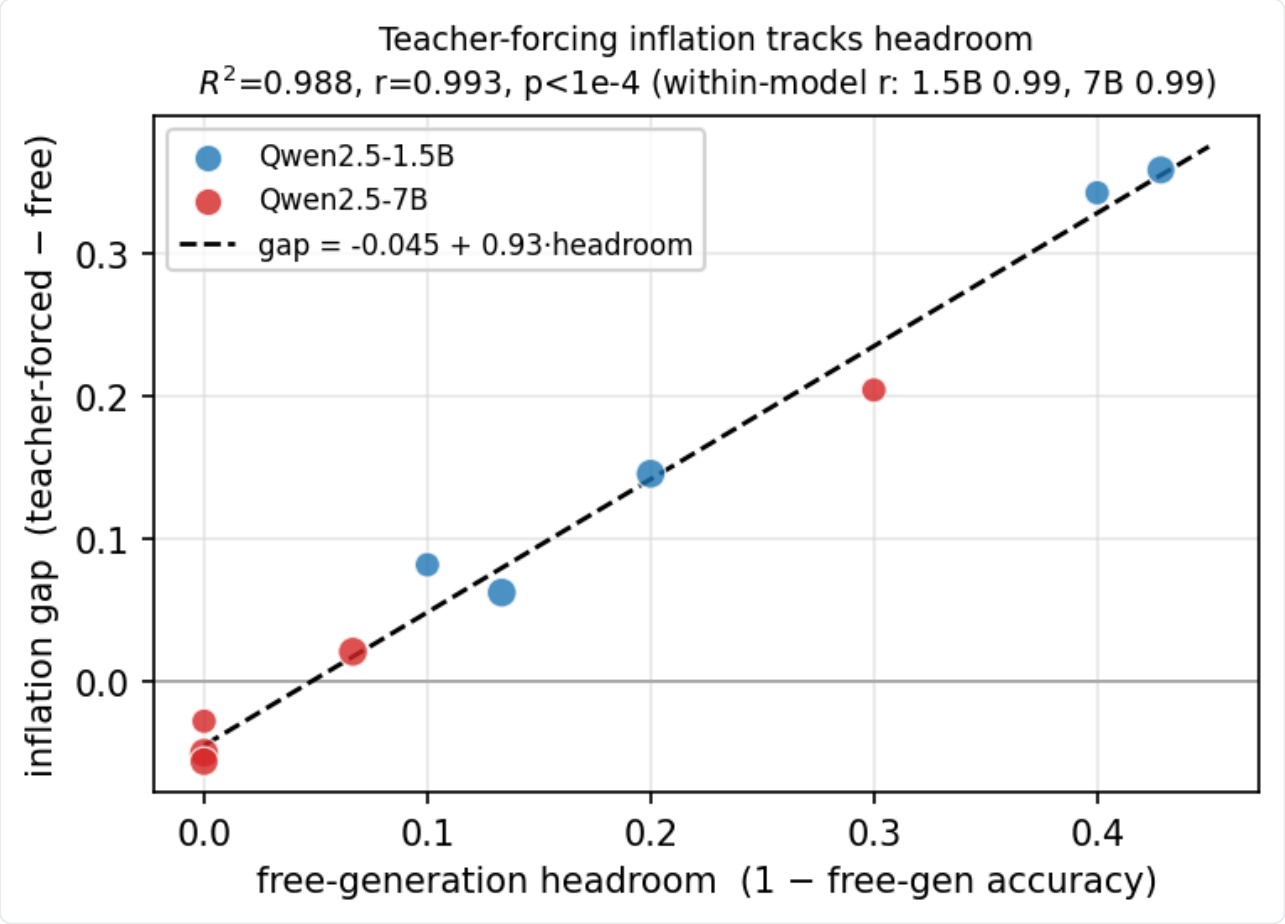
Hypothesis (H1). Teacher-forced token agreement overstates the capability a user obtains under free generation; therefore certificates must score with free generation.

Design. On a set of NL→SQL probes, we score the same model two ways: (a) free-generation execution accuracy (the model writes the whole query unaided; correct iff it executes to the gold result set), and (b) teacher-forced per-token agreement (the gold query is fed and we measure argmax agreement). We compare the two with a paired bootstrap. To test whether the effect is an artifact of one model or one difficulty level, we run a 2×2 — an easy probe set (n = 40) and a harder slice (n = 24; correlated subqueries, self-joins, multi-hop joins, HAVING over joined tables) — on both Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct (the 7B in 4-bit; quantization can only lower its quality, so it is a conservative setting for an inflation claim).

Result — the baseline operating point. On the easy set at 1.5B, teacher-forced agreement is 0.948 (95% CI [0.936, 0.961]) while free-generation execution accuracy is 0.85 ([0.725, 0.950]); the paired gap is **0.098, 95% CI [0.002, 0.213]** under set semantics (0.148, [0.034, 0.268] under stricter multiset scoring), n = 40. The gap is positive at the pre-registered significance level.

Result — the 2×2, reported honestly. The gap is *not* constant. At 7B the easy set saturates free generation (0.975), and the gap is consistent with zero (−0.018, [−0.053, +0.039]); on the hard slice the 1.5B gap grows to **+0.352 ([+0.157, +0.544])**, while the 7B hard gap is positive but not significant at n = 24 (+0.052, [−0.058, +0.186]). We pre-registered "the gap reappears at 7B (CI excludes 0)" and that criterion was **not met**; we report it as such rather than tuning the slice until it was. Crucially, the gap did not vanish at 7B because teacher forcing became honest — teacher agreement barely moved across all four cells (0.93–0.96) — but because free generation ran out of room to fail.

Result — the inflation law. That observation has a precise form. Conditioning each run on its per-probe difficulty tags yields ten operating points spanning free-generation *headroom* (defined as 1 − free-generation accuracy) from 0.00 to 0.43. The teacher-forcing gap is a near-deterministic linear function of headroom, **gap = −0.045 + 0.931 · headroom**, with weighted R^2 = 0.988, Pearson r = 0.993, and a seeded-permutation $p < 10^{-4}$. The same line fits *both* model sizes independently (1.5B r = 0.992, 7B r = 0.994). The slope near 1 means almost the entire shortfall between free generation and ceiling re-appears as teacher-forcing overstatement; the intercept near 0 means the two agree only at ceiling. The 7B non-significance is thus not a failure to replicate but the **low-headroom end of the same law**: teacher forcing misleads in proportion to how fallible the model is on the task.



The teacher-forcing inflation gap is a near-linear function of free-generation headroom, on one line across a 5× model-size range. Each point is a (model × difficulty) operating cell; marker size ∝ cell n.

Audit. A point estimate is not a mechanism, so we re-executed every failing free generation. The free-generation failures are *genuine model errors*: schema hallucinations (inventing a table that does not exist), aggregate misuse (an aggregate without a GROUP BY), a spurious LIMIT that drops valid rows, and convoluted-logic queries returning the wrong set. These are precisely the errors that *cannot occur* under teacher forcing, because feeding the gold token at every step prevents the model from ever emitting the spurious clause. Teacher forcing does not measure a slightly easier version of the task; it measures a different task in which derailment is impossible.

Implication. The certificate's efficacy and locality dimensions use free generation only — and the inflation law says exactly when this matters most: in the **fallible regime**, which is the regime where a capability is worth repairing. Where a model has effectively solved a slice (headroom ≈ 0) the two metrics agree, but there one would not patch at all. Teacher forcing is least trustworthy precisely where certification is most needed. The honest measuring stick is the foundation everything else rests on.

---

6. A Leaky Patch, Caught and Explained

Setup. We train a join-repair LoRA (rank 16) on join-requiring Spider queries. Training loss falls smoothly from 0.55 to 0.165 — by the signal most practitioners use, an unambiguous success. The base model's join execution accuracy is 0.380.

Certificate verdict: FAIL on three dimensions. Efficacy gain is +0.027 with a CI that includes zero (not significant). Locality drift is 0.18 — far above any reasonable threshold. Regression's general-QA suite drops 0.035. A patch every conventional signal calls a success is refused by the certificate.

Leak audit. We logged each probe's generated SQL and re-executed it. Of the patch's broken non-join queries, **72 of 73 are the same failure: the injection of an unwanted JOIN**. The patch learned Spider's canonical `FROM a AS T1 JOIN b AS T2 ON …` template and now applies it indiscriminately — for example, rewriting `SELECT PetID, weight FROM Pets WHERE pet_age > 1` as a gratuitous join against an unrelated table. Aggregation queries broke 35 and were fixed 8; schema- linking broke 38 and fixed 11; the join target itself moved by only +4 net. The patch did not learn "join planning"; it learned "emit joins," and the certificate's locality dimension is what exposes the difference. This audit table is the paper's central qualitative figure: the certificate's value is not the verdict alone but that the verdict is *explainable*.

Iterations and the non-isolation finding. Mixing non-join SQL into training (a locality anchor) fixes the in-domain leak — aggregation and schema-linking drift turn positive — but flattens efficacy (−0.013); a join-weighted mix at rank 32 recovers a *significant* efficacy gain (+0.080, CI [0.007, 0.153]) but still misses the 0.10 bar and still moves the SQL neighborhood. Across the family, every patch that gained efficacy moved the neighborhood, and every patch that held the neighborhood gained nothing. The conclusion is mechanistic: a low-rank update is not a surgical edit; it reshapes the whole text-to-SQL behavior at once. LoRA produces *significant but non-isolated* repair.

---

7. Calibrating the Certificate

A certificate is only as trustworthy as its thresholds. The locality limit had been set to 0.02 by fiat, and every measured patch landed between 0.06 and 0.44; worse, a brittle 8-item instruction probe moved by 0.125 on a *single* flip, repeatedly deciding verdicts on noise. We therefore calibrate the locality threshold from a measured noise floor, under a rule **fixed before any data was collected**.

The benign set. We measure how much each locality probe moves under perturbations that carry *no capability intent*: (Tier 1) random LoRA-shaped deltas at controlled norms — {0.25, 0.5, 1.0, 1.41}× of a reference patch's delta norm, three seeds each, no training; and (Tier 2) seed-variant retrains of the same recipe, whose pairwise disagreement is the run-to-run variation of an "equally good" patch.

The pre-registered rule. For each probe p, threshold_p = clamp(mean(benign_p) + 2·sd(benign_p), floor 0.02, cap 0.10); a probe whose raw value exceeds the cap is declared *unreliable* and must be replaced rather than used with a vacuous threshold. The efficacy bar is explicitly **not** calibrated — it is a product judgment, not a noise property, and lowering it after repeated failures would be the textbook Goodhart we are trying to prevent.

Findings. Two are load-bearing. First, **the destruction is structured, not a matter of magnitude**: a random delta at the *merged* norm (the 1.41× tier) is harmless on the SQL slices, even though a real merge at that same magnitude is catastrophic (§9). The damage lives in specific learned directions, not in the size of the perturbation. Second, the apparent *seed lottery* in side effects is partly a small-sample artifact: at adequate resolution, the instruction-following cost of SQL training is *systematic* (≈ −0.07), not a coin flip. A corollary makes resolution a design constraint: a threshold of 0.02 requires a probe of at

least 51 items, because one flip out of 50 already exceeds it; our 48-item probe was coarser than its own threshold, and we enlarge it to a 200-item seeded sample (one flip = 0.005) before trusting any further locality verdict.

Consequence. With calibrated thresholds (and a coarse-grained capability — text-to-SQL as a whole, with non-SQL locality), the efficacy dimension passes for the first time in the project (+0.21–0.23 across seed variants). The certificate is *satisfiable*; the remaining failures are real.

---

8. The Patch-Mechanism Wall

8.1 The boundary

With a calibrated, resolution-correct, deliberately lenient coarse certificate, three independently trained aggregation patches all clear efficacy by a wide margin but all fail locality on a consistent instruction-following regression of ≈ −0.07. The finer probe that could have rescued a borderline candidate instead *revealed* that the borderline was noise and the harm is real and reproducible. The bottleneck is not the certificate — calibrated, resolution-correct, robust — but the **patch mechanism**: LoRA cannot, at this scale, produce a capability gain isolated enough to pass.

8.2 The certifiable corner

We then probe the unexplored gentle end of the efficacy–locality frontier (rank 4–8, one epoch, low learning rate). A rank-8 gentle patch reaches the certifiable corner: efficacy +0.20, locality drift collapsed to exactly 0.02, regression clean except for general-QA at exactly the −0.02 boundary, where it fails by a floating-point epsilon (a count of 4/200). It is one recipe tweak from a clean pass. A rank-4 patch is too gentle and loses efficacy. The lesson is the inverse of §8.1's pessimism: a *certified* LoRA patch is not impossible, it is marginal — the corner exists.

8.3 An integrity decision, recorded

The rank-8 near-miss failed only by a float epsilon, on a real implementation bug (a regression of exactly the allowed amount should pass; strict `<` plus float representation rejected it). We chose to honor the pre-registered "this is the final attempt for these candidates" commitment: both gentle candidates stand as recorded FAILs, no certified patch is claimed, and the bug is fixed *forward only* — with a test asserting the committed near-miss artifact keeps its recorded FAIL. We report this because the discipline is the point: a framework whose verdicts can be quietly massaged at the boundary certifies nothing.

---

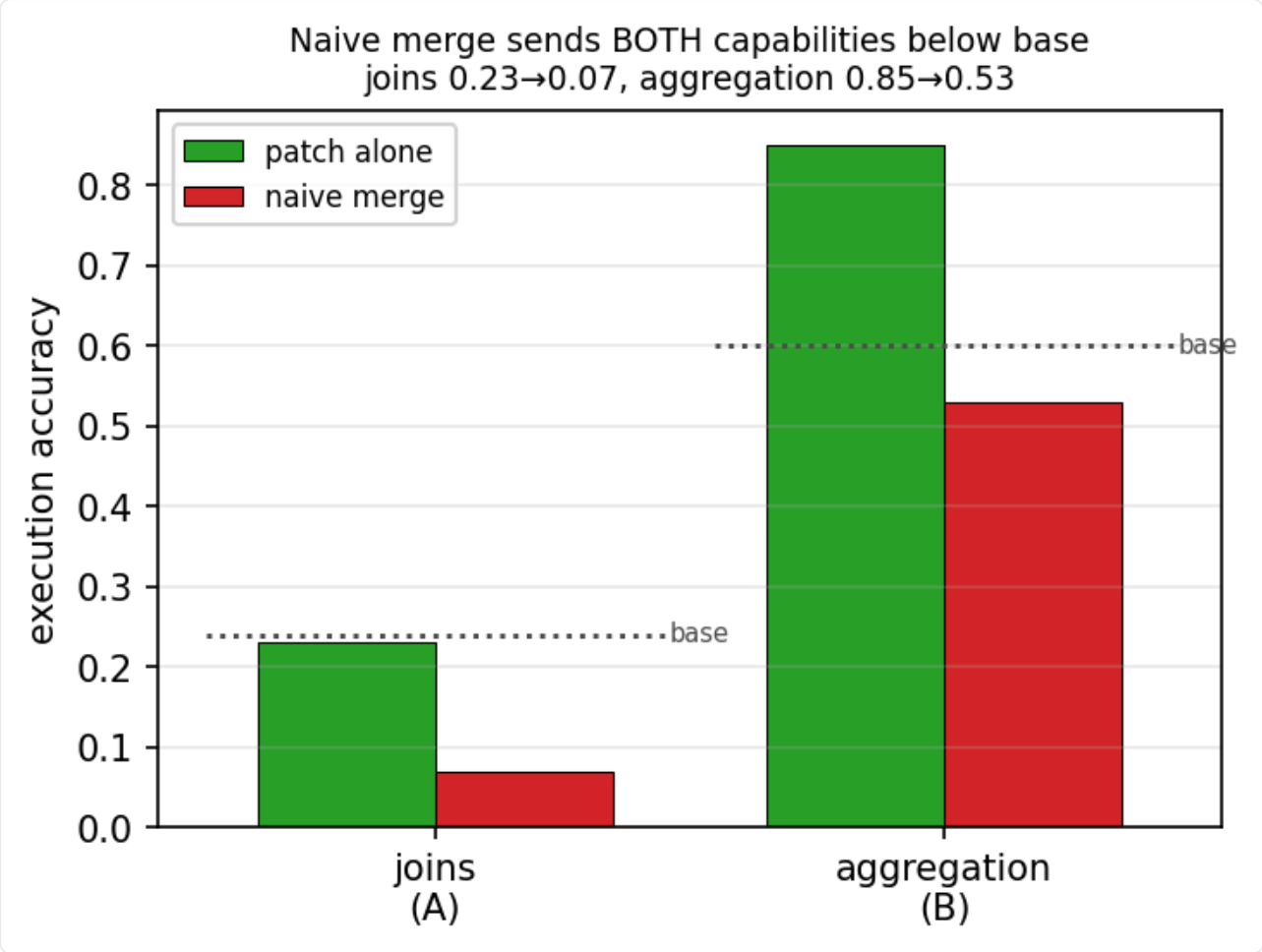
9. Composition Breaks Certificates

Hypothesis (H2). Co-installing two individually certified patches degrades a dimension that held for each alone.

Design. Two pure single-capability patches — A on joins, B on aggregation — each strong in isolation. We naively merge them (sum of deltas) and re-certify each capability on the merge. We use *pure*

patches deliberately: H2 is about whether *independently* trained patches interfere, so anchoring one on the other's capability would pre-entangle them.

Result. Alone, B is the strongest patch in the entire project: aggregation 0.60 → 0.85, a *significant* +0.25. Merged, **both capabilities collapse below the base model**: joins fall from a base of 0.24 to 0.07 (significant, CI excluding zero), and aggregation falls from 0.85 to 0.53, below the base of 0.60. The merge does not merely fail to preserve B's certificate; it annihilates B's gain *and* damages the base capability. A certificate is therefore **not composition-preserved** under naive merge — the second half of the central claim, demonstrated.



Each capability is strong as a standalone patch (green) but, under naive merge (red), both fall below the unpatched base (dotted) — the certificate is not composition-preserved.

Mechanism preview. The damage is destructive interference: summing two broad low-rank updates, each of which reshapes the whole SQL behavior (§6), produces a combined update that is worse than either and worse than nothing. This is consistent with §7's magnitude-refutation: a random delta at the merged norm is inert, so it is the *structure* of the two learned directions, not their combined size, that destroys.

---

10. Predicting Conflict From the Patch Weights

Hypothesis (H3). The composition conflict is predictable, before merging, from features computed from the two patches' weights alone — with no co-certification.

A reversed prediction, recorded. We initially predicted that *high* subspace overlap would cause conflict (two patches writing to the same place collide). A first measurement refuted this: the catastrophic A–B pair of §9 has *low* overlap (principal-angle 0.174 versus a chance baseline of ≈ 0.073 for random rank-32 subspaces in this architecture) and near-zero directional alignment, yet its merge is catastrophic. We record the refutation and revise: the live hypothesis becomes that conflict is governed by norm structure and interference, with overlap entering with the *opposite* sign to intuition.

Features (offline, from weights). Per layer, the LoRA delta is $\Delta W = BA$; we compute the principal-angle subspace overlap and cosine alignment between the two patches' deltas, the Frobenius norms $\|\Delta X\|$, $\|\Delta Y\|$, their sum, and an **interference ratio** $\|\Delta X + \Delta Y\| / (\|\Delta X\| + \|\Delta Y\|)$ — below one when the deltas cancel (destructive), near one when additive. No GPU and no forward pass are needed.

Targets (cheap). For nine declared pairs spanning cross-capability and same-capability, seed-variant and rank-variant combinations, we measure the worst-case efficacy degradation of the naive merge with efficacy-only passes against a fixed seeded sample.

Results. Univariate Spearman correlations of the features with signed degradation are strong: subspace overlap **+0.85** (confirming, with the reversed sign, that *more overlap is safer*), combined norm **−0.82**, interference ratio **+0.77** (more cancellation, more harm). Two sharper findings emerge. First, **"same capability is safe" is false**: two full-rank aggregation patches collapse each other (−0.32, −0.25) just as cross-capability pairs do (−0.26 to −0.33). Second, the governing variable is **norm dominance**: the only safe merges pair a strong patch with a tiny one (−0.04, +0.01), where the merge is ≈ the dominant patch and the subordinate is harmlessly swamped; whenever the two norms are comparable — two rank-32 patches, or two gentle ones — both collapse. Conflict is destructive interference between deltas of *comparable magnitude*; a swamping patch survives.

Verdict: inconclusive, by our own criterion. A pre-registered leave-one-pair-out linear predictor on the full feature set overfit badly (six features, nine points: cross-validated Spearman 0.017), while overlap alone reached 0.583 — just under the pre-registered 0.6 bar. We therefore report H3 as **directionally strong but formally inconclusive**, and we own the pre-registration flaw: six features for nine pairs is over-parameterized, and the missing feature the mechanism points to is an explicit norm-dominance term (min/max $\|\Delta\|$). A deployable predictor needs more pairs and that feature; we leave it to a follow-on. The directional signal is nonetheless immediately actionable: an install policy can refuse a merge of comparable-magnitude, low-overlap, high-cancellation patches without ever paying for co-certification.

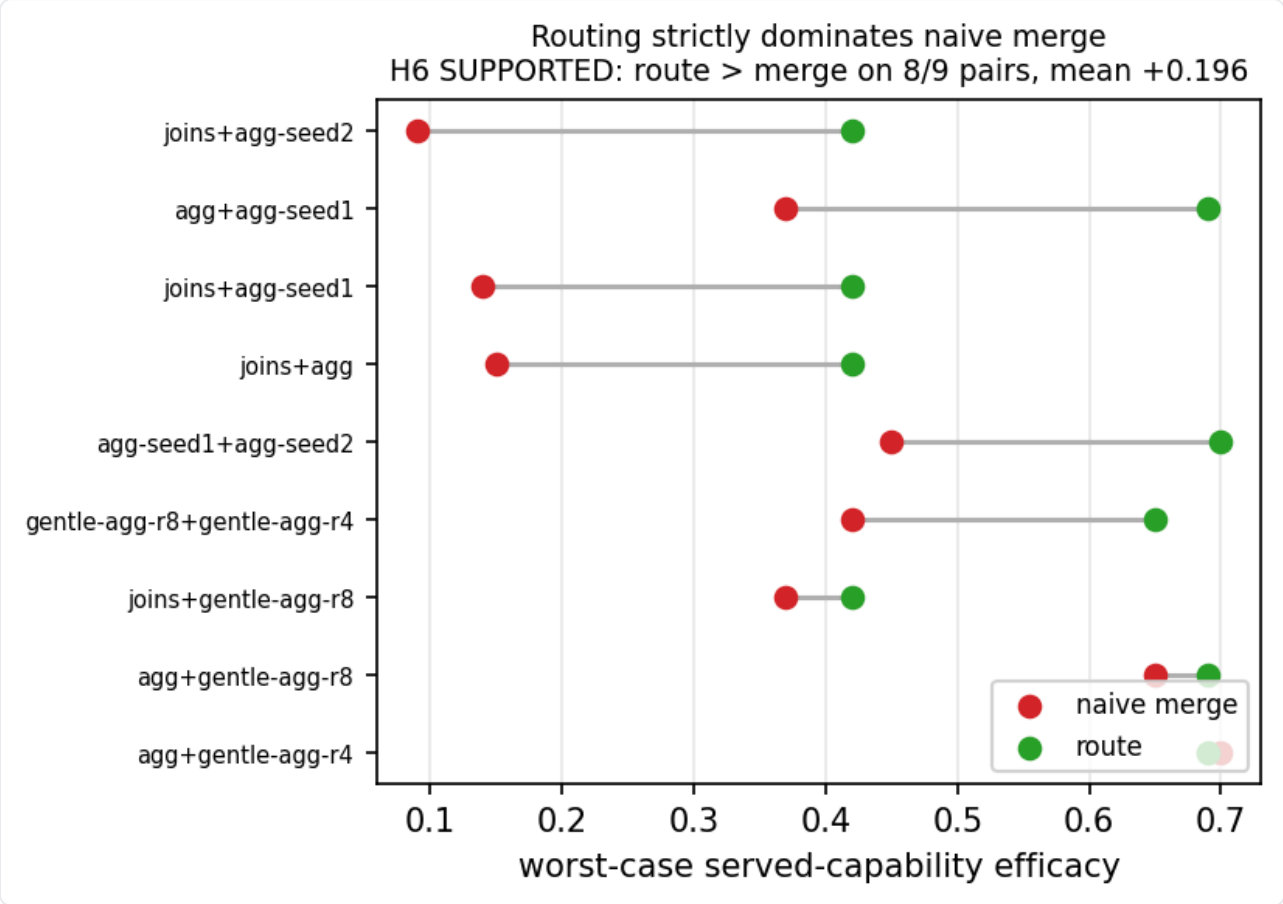
---

11. Routing Beats Merging

Hypotheses. H6: routing (keep the patches separate; serve each capability with its own patch) beats naive merge. H5: a weight-based policy that *chooses* between merging and routing beats naive merge.

Design (offline). From the §10 grid we compute, per pair, each operator's worst-case served-capability efficacy. *Route* serves each capability with its dedicated patch run alone — by construction free of interference. *Naive merge* uses the measured merged efficacy. A weight-based policy merges only when one patch dominates (min/max norm < 0.5, an a-priori threshold declared before computing, not fitted to the nine outcomes) and routes otherwise. Criteria are pre-registered.

Results. H6 is supported decisively: routing beats naive merge on 8 of 9 pairs, mean margin +0.196. Naive merge sends both capabilities below base; routing recovers ≈ 0.2 of execution accuracy by simply not combining the weights. The single non-win is a strong-plus-tiny pair where merge and route tie. **H5 is *not* supported — and the reason is the finding.** The weight-based policy is never worse than naive merge (the safety property holds), but it never *beats* routing, because **naive merge is strictly dominated**: route ≥ merge on every pair. There is no regime in which merging wins, so there is no niche for a selective-merge policy. The optimal install policy collapses to a one-liner: *always route, never merge.* The weight-based policy earns its keep only under a hard deployment constraint that forbids keeping multiple adapters, where it correctly avoids the catastrophic comparable-magnitude merges.



Routing strictly dominates naive merge: per pair, the worst-case served-capability efficacy under merge (red) vs route (green). Route ≥ merge on every pair.

This closes the composition arc. Naive merge is catastrophic (§9); the catastrophe is readable from the weights (§10); and the install operation that fixes it is routing, not a cleverer merge (§11).

12. A Second Domain: Tool-Call Certification

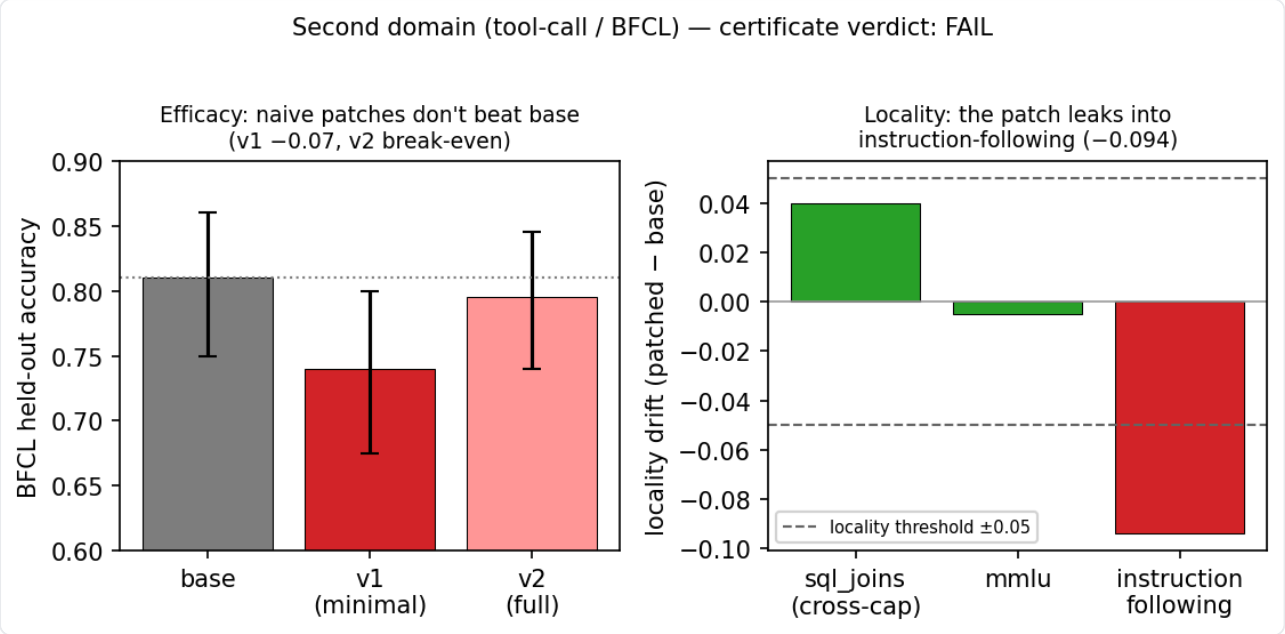
Everything above is text-to-SQL. To test whether the framework is SQL-shaped or a general instrument, we certified a second, unrelated capability — **tool/function-call generation** — on a recognized external benchmark, the Berkeley Function-Calling Leaderboard (BFCL, v4 `simple` Python category, pinned by commit). The capability passes the same selection gate: the metric is a faithful subset of BFCL's AST checker (exact function name, no hallucinated arguments, each argument value matching one of the benchmark's accepted values, optional arguments omittable), scoring is greedy free generation, and the eval is a held-out split disjoint from the training pool by construction.

The base model is already strong, and its failure is isolated. On 200 held-out probes the base model scores 0.81; function *selection* is essentially perfect (198/200 right function, zero wrong-function errors), and all of the loss is in argument *values*. This is an ideal patch target: measurable, repeatable, and isolated to one sub-behavior.

Two patch attempts, both refused — and the certificate explains why. A first LoRA, trained on minimal canonical-argument targets, *reduced* held-out accuracy to 0.74 (paired gap −0.07): the targets systematically omitted optional arguments, so they were shorter than the model's natural calls, and the adapter learned an argument-*dropping* bias that it over-applied to required arguments too. This is a sharper version of the §6 lesson — here the patch fails on *its own objective*, and only the free-generation paired measurement reveals it. A second attempt that fixed exactly that one variable (fuller targets including optional arguments) recovered most of the loss but landed at break-even (0.795–0.81, gap not distinguishable from zero). Naive supervised fine-tuning on benchmark answers fixes roughly as many held-out probes as it breaks. The efficacy dimension **fails** for both: the patch-mechanism wall of §8 recurs in a second domain.

The certificate also catches a leak the patch was not suspected of. The full certificate for the break-even patch fails on a second dimension as well: locality. Its declared neighborhood includes a cross-capability probe (Spider join execution accuracy), MMLU, and instruction-following. Join accuracy moves +0.04 and MMLU −0.005 (both within noise), but **instruction-following drops 0.094** — a real collateral regression in a behavior the tool-call patch has nothing to do with. This failure is robust to the (provisional, pending-calibration) locality threshold: 0.094 exceeds any plausible calibrated floor. So the patch is simultaneously **useless on its target and harmful off it**, and the per-dimension certificate states both facts where training loss and a single accuracy number would have shown a clean-looking result.

A repeatable cross-capability effect. The instruction-following regression is not a one-off: SQL capability training degraded instruction-following by ≈0.07 (§7), and tool-call training degrades it by 0.094. Two unrelated capability patches, the same collateral victim — evidence that capability SFT generically erodes general instruction-following, exactly the kind of effect a locality dimension is meant to surface and an averaged score would hide. The framework transfers: a different capability, a different external benchmark, the same machinery, and the certificate again refuses a patch every conventional signal would accept.



Second domain (tool-call / BFCL). Left: naive patches do not beat the base (v1 reduces accuracy; v2 is break-even). Right: the break-even patch nonetheless leaks, dropping instruction-following by 0.094 past the locality threshold — the certificate FAILs on efficacy and locality.

---

13. Discussion

The certificate is the validated contribution; the registry is the open question. Across the study, the certificate did its job at every step — it refused a loss-looks-great patch, it caught composition collapse, it was made satisfiable by calibration, and it held the line at the boundary. What remains genuinely open is whether *any* cheap patch mechanism can produce a *certified* patch: LoRA reaches the certifiable corner but lands on its edge. This re-frames "a registry of certified patches" from a deliverable we are failing to hit into the precise research question our validated instrument now lets us ask — and points to isolation-by-construction mechanisms such as null-space- constrained editing, applied at capability rather than fact granularity, as the natural next patch family to put through the certificate.

Composition has a structural answer and a predictive one. The structural answer — route, do not merge — is simple and strong. The predictive answer — conflict is readable from norms, overlap, and interference — lets an install policy refuse or route *before* paying for co-certification, which is the expensive step. Together they are the composition-preservation product the framework set out to build.

Honesty as a load-bearing component. Pre-registration and an append-only journal let us report a near-miss as a FAIL (§8.3), a reversed prediction as a refutation (§10), and a probe-resolution defect as a forward-only fix (§7) — without spending credibility. A certification framework whose verdicts can be quietly tuned certifies nothing; the discipline is not adjacent to the contribution, it is part of it.

---

14. Limitations

- **Scale coverage is partial.** The measuring-stick law (§5) is established across 1.5B and 7B, but the certification, calibration, and composition results (§6–§11) are on Qwen2.5-1.5B-Instruct only. The

non-isolation of LoRA and the magnitude of composition collapse may differ at 7B+; those runs are in progress, not yet recorded.

- **Two domains; composition on one.** Certification now covers two capabilities — text-to-SQL on Spider and tool-calls on BFCL (§12) — but the composition results (§9–§11) are SQL-only, and both domains are at the 1.5B scale. A third domain and cross-domain composition (SQL × tool-call) would further test generality; the BIRD generalization slice the certificate schema was designed around is also deferred.
- **Small-n conflict analysis.** The §10 predictor rests on nine pairs sharing a single joins patch; it is directional, not a deployed model, and a clean fit could partly be same-vs-cross-capability classification in disguise — which we flag per pair.
- **Oracle routing.** §11 assumes a correct query→capability router; a learned router is a separate component we specify but do not build.
- **No certified PASS yet.** The strongest patch is a boundary near-miss. We claim the certifiable corner exists, not that we have populated the registry.
- **Concurrency.** §10 overlaps with concurrent work; we differentiate but cannot claim priority on merge-predictability per se.

---

15. Conclusion

Capability patches cannot be trusted on the strength of training loss or a target metric: they leak, and they do not survive composition. We make the **certificate** — a calibrated, per-dimension, free-generation behavioral statement with composition preservation — the unit of trust. On text-to-SQL we show the certificate catches a patch every conventional signal endorses, that its thresholds must and can be calibrated from a measured noise floor, that naive composition drops two strong patches below base, that the resulting conflict is predictable from the patch weights, and that the safe install operation is routing rather than merging. The patch is an artifact; the certificate, and the proof that it survives composition, is the product.

Appendix A — Reproducibility

Base model pinned by commit SHA; greedy decoding; bootstrap CIs at fixed seed; seeded random eval sampling (not prefixes). Each experiment is a single runnable entrypoint writing a schema-validated JSON artifact, with resumable per-pass checkpointing. The complete pre-registration-and-results journal, all certificate artifacts, the calibration data, the conflict grid, and the install-policy evaluation are kept alongside the code and available on request. Unit tests cover the pure-logic core (scoring, verdict boundary semantics, calibration rule, predictor, policy).

Appendix B — Pre-registration log (selected)

- Locality calibration rule (clamp(mean + 2sd, 0.02, 0.10); unreliable above cap) — declared before calibration data.

- text-to-SQL@2.0 re-cert protocol (fresh eval seed, any-PASS-counts, last amendment for these candidates) — declared before the run.
- Probe-resolution fix (instruction probe n→200 seeded; thresholds unchanged) — declared as a measurement fix, final amendment.
- H3 predictor verdict (LOO $|\rho| \geq 0.6$ and overlap+norm beats overlap-alone; refuted if < 0.3) and H5/H6 criteria (a-priori dominance threshold 0.5) — declared before computing.

Appendix C — Calibrated thresholds (cert-spec, artifact-level)

Locality, from the benign placebo floor: joins ≈ 0.030, aggregation ≈ 0.047, schema-linking ≈ 0.040, MMLU ≈ 0.020, instruction-following ≈ 0.020 (with the enlarged n=200 probe). Efficacy bar 0.10 (with CI excluding zero), uncalibrated by design. Regression: perplexity ratio ≤ 1.03; general-QA drop ≤ 0.02. Boundary values pass, float-tolerant.

References

A full bibliography will accompany a later revision. Works referenced in the text include: *The Mirage of Model Editing*; *Are We Evaluating Edit Locality Properly?*; *Provably Safe Model Updates*; *Demystifying Mergeability*; the ROME / MEMIT model-editing lineage; Task Arithmetic, TIES, and DARE weight-merging; LoRA; AlphaEdit (null-space-constrained editing); and the Qwen2.5, Spider, BFCL, MMLU, and WikiText models and datasets.