VERIFY WHAT YOU CAN PARSE: PROMPT-GATED RESIDUAL RERANKING FOR REWARD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reward models must assess both semantic quality and literal instruction compliance, although constraints such as exact counts, required phrases, and output formats are often more reliably evaluated by explicit programs. We introduce SPECGATE, a post-hoc reranking layer for frozen reward models. A prompt parser restricts intervention to supported constraints, an executable checker measures literal compliance, and a 28.2K-parameter ensemble predicts complementary soft compliance signals. A spread gate suppresses learned corrections when their evidence is weak. Using a configuration fixed before the reported replay, SPECGATE improves official RewardBench 2 scores from 82.21 to 85.44 on Llama-3.1-8B and from 77.83 to 81.38 on Qwen3-8B. The aggregate changes are confined to Precise Instruction Following, where the method produces 31 and 34 net corrections, respectively. These corrections satisfy $\Delta_{\text{overall}} = \Delta_{\text{PreciseIF}}/6$, matching the benchmark's six-domain macro-average. Paired bootstrap confidence intervals are $[2.08, 4.38]$ and $[2.40, 4.79]$ percentage points, with exact McNemar $p < 1.3 \times 10^{-7}$. Ablations show that the executable checker contributes most of the improvement, while the learned component provides a smaller, model-dependent benefit. These results support selective verification for candidate reranking when prompts expose mechanically checkable requirements; they do not imply a general replacement for semantic reward modeling.

1 INTRODUCTION

A reward model is often asked to solve two different problems with one scalar. It must judge semantic quality—correctness, relevance, safety, and style—and it must notice literal contract violations such as "use exactly four numbers," "return only JSON," or "end with `Thank you.`" The first problem benefits from broad language understanding. The second is frequently easier for a parser and a short program. Nevertheless, standard Bradley–Terry reward models (Bradley & Terry, 1952; Ouyang et al., 2022) learn both through the same global parameter update.

This coupling is costly near a strong anchor. A broad update can repair a small family of instruction following errors while perturbing thousands of comparisons that were already correct. It also spends model capacity on operations such as counting words and checking delimiters, for which deterministic computation is more reliable. The right abstraction is therefore not always "train a better reward model." It can instead be:

detect when a narrow verifier is competent, and authorize a local reranking only there.

We instantiate this idea with SPECGATE, a prompt-gated residual reranker around frozen Skywork-Reward-V2 anchors (Liu et al., 2025a). A deterministic parser recognizes 38 families of checkable constraints. On recognized prompts, a tiny MLP called CountResidual combines the parsed specification with response counts and satisfaction proxies. An executable checker independently scores literal compliance. On every other prompt, the final ordering is exactly the frozen anchor's ordering.

The resulting system has two useful properties. First, it has a small blast radius: the parser fires on 167 of 1,865 RewardBench 2 prompts, including 159 of 160 Precise-IF prompts and only eight

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.

prompts outside that subset. Second, its failure modes are inspectable. We can remove the parser gate, remove the residual confidence gate, zero each feature family, or train on synthetic templates alone and observe the effect directly.

Results. Under the official RewardBench 2 scoring code at commit `e4050493415ed7183a38d1778e148c49904f6f05`, the fixed gated system improves exactly reproduced Llama-3.1-8B and Qwen3-8B anchors by $+3.23$ and $+3.54\,\mathrm{pp}$. These are system-level gains relative to the in-repository anchor caches, not claims that a newly trained scalar reward model improved by the same amount. CountResidual by itself improves the anchors by $+1.77/+2.19\,\mathrm{pp}$ in benchmark-internal component analysis; the live checker is the larger contributor. The best analyzed Qwen configuration reaches $+3.65\,\mathrm{pp}$ and sets the residual weight to zero, an important negative result that we report rather than hide.

What the result means. The method is an instance of selective computation, not a claim that symbolic rules replace semantic reward models. The anchor remains responsible for broad preference judgment; the verifier acts only where the prompt itself exposes a decidable contract. The resulting gain is large because the specialist is accurate on its support and exactly inert elsewhere. This interpretation is stronger than attributing the result to a generic learned head: it predicts the domain localization, the gate ablation, and the observed disappearance of the learned residual at the Qwen optimum.

Auditability. A commit-pinned replay also corrected the identity and evaluation of an earlier draft: archived 87.21/82.68 scores came from this CountResidual/checker stack, not an ordinal reward head, and used a row-wise rather than official paired Ties metric. We retain those records in Appendix E, while all headline values use official scoring.

Contributions.

- We derive selective reward correction from support mismatch and a normalization scale paradox. The resulting prompt and spread gates provide exact off-support invariance and vanishing-evidence continuity.
- We introduce SPECGATE, combining a frozen semantic anchor, a 28.2K-parameter three-head CountResidual ensemble, and an executable verifier without changing the anchor weights.
- We connect mechanism to metrics: net corrections predict the macro-average gain exactly; paired transitions and confidence intervals establish that improvements dominate harms; gate, feature, and data ablations explain why.
- We provide an evaluator-corrected, hash-tracked reproduction and a claim ledger that separates fixed results, benchmark-internal mechanism analysis, and archival evidence.

2 RELATED WORK

Reward models, alignment, and evaluation. Pairwise reward learning under Bradley–Terry models has scaled from agent trajectories to language generation (Bradley & Terry, 1952; Christiano et al., 2017; Ziegler et al., 2019; Stiennon et al., 2020). Modern RLHF systems train assistant policies against learned preferences (Bai et al., 2022; Ouyang et al., 2022), while DPO optimizes the corresponding preference objective without an explicit reward model (Rafailov et al., 2023). Curated resources such as HelpSteer2 and Skywork-Reward-V2 strengthen scalar reward models (Wang et al., 2024; Liu et al., 2025a). RewardBench, RM-Bench, and RewardBench 2 expose complementary domain, subtle-content, style, and best-of-n failures (Lambert et al., 2025; Liu et al., 2025b; Malik et al., 2025); reward overoptimization further cautions against broad changes near a strong anchor (Gao et al., 2023).

Rubrics, specifications, and verifiable evaluation. LLM-as-a-judge provides scalable but biased open-ended evaluation (Zheng et al., 2023). Auto-Rubric infers query-specific criteria (Xie et al., 2025), and prompt-level reward specifications make requirements explicit (Weng et al., 2026). IFEval, FollowBench, and InFoBench test instruction compliance with executable or decomposed constraints

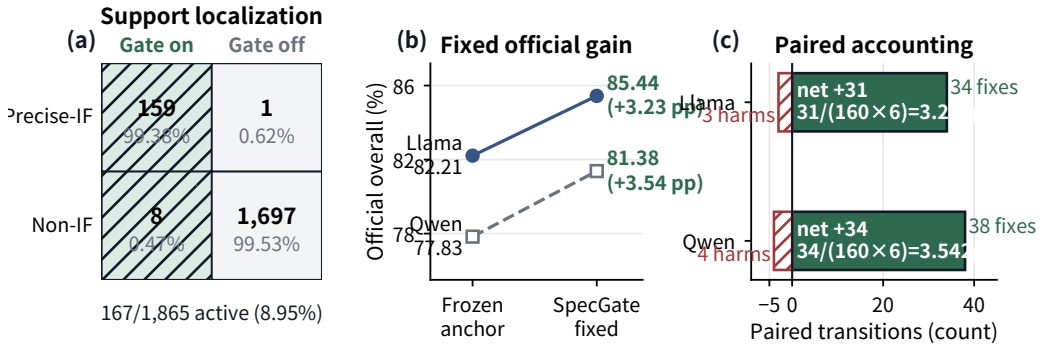

Figure 1: **Selective verification in one view.** A broad semantic anchor handles every prompt; the verifier activates on only 8.95% of prompts. The fixed system changes only Precise-IF and raises official overall by $3.23/3.54$ pp on Llama/Qwen. The final asset should combine the support argument, domain-localized deltas, and paired fix/harm counts.

(Zhou et al., 2023; Jiang et al., 2024; Qin et al., 2024). SPECGATE combines an exact prompt gate, a learned count residual, and a direct checker, then quantifies when each component is useful.

Verification, tool feedback, and selective prediction. Learned verifiers improve candidate selection in mathematical reasoning (Cobbe et al., 2021; Uesato et al., 2022; Lightman et al., 2024); self-refinement and tool-interactive critique instead feed evaluator signals back into generation (Madaan et al., 2023; Gou et al., 2024). Our reranker does not regenerate: it composes a broad learned judge with narrow executable predicates. Its support gate is related to selective prediction and learning to defer (Geifman & El-Yaniv, 2017; Madras et al., 2018), but uses a deterministic competence declaration to guarantee exact off-support invariance.

3 METHOD

3.1 PROBLEM FORMULATION

For a prompt x, let $\mathcal{Y}(x) = \{y_1, \ldots, y_n\}$ be the candidate set and let $a_j = r_0(x, y_j)$ be the score from a frozen anchor reward model. We seek a reranking score s_j. Our setting differs from training a standalone scalar reward in two ways. We assume the full candidate set is available, and we permit a deterministic verifier at inference time. This fits best-of-n selection, data filtering, and benchmark reranking. It is not directly a drop-in reward for online PPO, because candidate-wise normalization makes s_j depend on $\mathcal{Y}(x)$.

Sparse competence. Let $g_p(x) \in \{0, 1\}$ indicate that the prompt parser recognizes at least one supported constraint. The design contract is

$$g_p(x) = 0 \quad \Longrightarrow \quad s_j = a_j \;\; \forall j. \tag{1}$$

Thus a specialized signal cannot alter a comparison outside its declared competence region. This is stronger than hoping a learned gate generalizes to unrelated domains: it is an exact implementation invariant.

Why a learned residual as well as a checker? An executable checker is precise when parsing and implementation are correct, but hand-written rules can be brittle to linguistic variation. A learned residual can combine softer evidence, such as how far a word count is from its target or whether several count statistics jointly support the constraint. We therefore test both signals and their interaction rather than assuming either one is universally superior.

Design rationale. The architecture follows from the geometry of the intervention rather than from a search over reward heads. We derive the two gates and the expected metric signature before presenting the implementation.

3.2 SUPPORT GATING

Let $\mathcal{V} = \{x : g_p(x) = 1\}$ be the prompts on which the verifier claims competence, and let $\rho = \Pr(x \in \mathcal{V})$. For an ordinary per-prompt ranking utility U, define the gain of adding a specialist correction as

$$G = \mathbb{E}[U(s) - U(a)] = \rho G_{\mathcal{V}} + (1 - \rho) G_{\neg\mathcal{V}}. \tag{2}$$

A constraint specialist has a reason to make $G_{\mathcal{V}} > 0$, but no such reason outside $\mathcal{V}$. If its average off-support damage is only ϵ, then a net gain requires $G_{\mathcal{V}} > (1 - \rho)\epsilon/\rho$. In our evaluation, the parser activates on $167/1{,}865 = 8.95\%$ of prompts, so one unit of off-support damage requires more than 10.2 units of on-support improvement to offset. A global correction is therefore statistically mismatched to the problem even when its off-support error is small.

Proposition 1 (support safety). The piecewise rule in Eq. 1 has zero pairwise regret relative to the anchor on $\neg\mathcal{V}$. *Proof.* For every $x \notin \mathcal{V}$, $s_j = a_j$ for all candidates, so every pairwise ordering is identical. □

This elementary guarantee is the main reason to use a deterministic prompt gate rather than a soft domain classifier. It converts specialization from a global average-risk bet into a local one.

3.3 SPREAD GATING

Candidate-wise standardization is useful because anchor and verifier scores have incomparable units, but it destroys residual magnitude. For any nonconstant vector v, scalar b, and $\varepsilon > 0$ satisfying $\varepsilon \,\mathrm{sd}(v) > \epsilon_0$, where ϵ_0 is the numerical standard-deviation floor,

$$\mathrm{z}(b\mathbf{1} + \varepsilon v) = \mathrm{z}(v). \tag{3}$$

Thus, as a learned residual becomes arbitrarily close to a constant, its standardized ranking signal retains full scale until it crosses the implementation floor and drops abruptly to zero. A weak just-above-floor prediction can therefore receive the same nominal weight as a confident prediction.

Proposition 2 (vanishing-evidence continuity). With $\gamma(q) = \min(1, \mathrm{range}(q)/\tau)$, the learned correction $\gamma(q)\,\mathrm{z}(q)$ converges to zero as $\mathrm{range}(q) \to 0$ for a fixed finite candidate set. *Proof.* For n candidates, each standardized coordinate has magnitude at most $\sqrt{n-1}$; the numerical floor can only reduce it. Hence $\|\gamma(q)\,\mathrm{z}(q)\|_\infty \le \sqrt{n-1}\min(1, \mathrm{range}(q)/\tau) \to 0$. □

The prompt gate and spread gate solve different problems: the former controls *where* the specialist may act; the latter controls *whether its learned evidence is strong enough* to act.

3.4 METRIC IMPLICATION

RewardBench 2 reports the unweighted mean of six domain scores. If SPECGATE is genuinely local and changes only Precise-IF, then

$$\Delta\mathrm{Overall} = \frac{1}{6}\Delta\mathrm{PreciseIF}. \tag{4}$$

This is not merely an interpretation after seeing the result. It is the metric-level prediction of Eq. 1. The fixed Llama system improves Precise-IF by 19.375 pp, predicting $19.375/6 = 3.229$ pp overall. The fixed Qwen system improves it by 21.25 pp, predicting $21.25/6 = 3.542$ pp. Both equal the measured gains up to rounding.

The derivation yields three empirical tests:

1. removing the prompt gate should spread damage across unrelated domains;
2. removing the spread gate should amplify weak learned differences;
3. a complete executable checker should dominate its learned approximation, while the residual should help only where the checker leaves useful margin.

Sections 4 and 4.4 test each prediction.

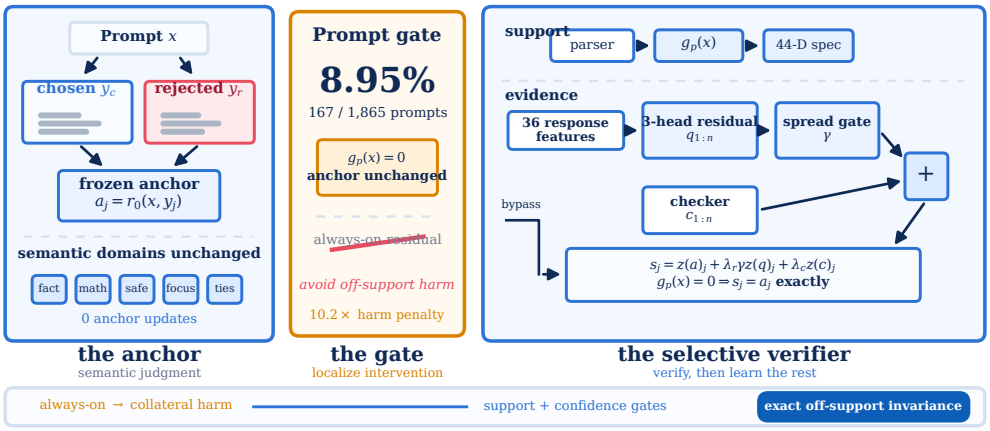

Figure 2: **SPECGATE architecture.** The frozen anchor handles general semantic quality. A prompt-only parser authorizes two specialized signals only for recognized constraints. The spread gate attenuates uncertain learned residuals; the direct checker remains active.

3.5 PROMPT SPECIFICATION AND RESPONSE STATISTICS

The parser maps x to a 44-dimensional specification vector. It includes a has-constraint bit, a normalized constraint count, a one-hot representation over 38 families (including an `other` bucket), and four numerical or phrase-related fields. The supported families cover counts, exact phrases, prefixes and suffixes, JSON-only responses, lists, punctuation, nesting, per-sentence rules, restricted answer formats, and related literal requirements.

For each candidate, we concatenate four feature groups:

$$f(x, y_j) = [\, p(x) \in \mathbb{R}^{44}; c_{\text{raw}}(y_j) \in \mathbb{R}^{18}; i(x, y_j) \in \mathbb{R}^{10}; e(x, y_j) \in \mathbb{R}^{8} \,] \in \mathbb{R}^{80}. \tag{5}$$

The response-only counts include normalized numbers, words, names, pronouns, lines, nesting depth, palindromes, punctuation, emoji endings, list markers, and unique words. Interaction features compare these counts with requested targets. Eight extra proxies cover the less common sentence-position, sentence-balance, word-step, vowel, alliteration, and multiple-choice families. No benchmark subset label or preference label is an input.

3.6 COUNTRESIDUAL

CountResidual is a two-hidden-layer MLP,

$$h_\theta(f) = W_3\,\text{GELU}(W_2\,\text{GELU}(W_1 f + b_1) + b_2) + b_3, \tag{6}$$

with width 64 and 9,409 parameters. Its last layer is zero-initialized. Each response has a checker target $c(x, y) \in [0, 1]$, and the network predicts the centered value $c(x, y) - 0.5$ using weighted MSE:

$$\mathcal{L}_{\text{res}} = \mathbb{E}_{(x,y)}\left[(1 + 2g_p(x))\big(h_\theta(f(x, y)) - (c(x, y) - 0.5)\big)^2\right]. \tag{7}$$

We train three prompt-group-disjoint heads and average them:

$$q_j = 0.5 + \frac{1}{3}\sum_{k=1}^{3} h_{\theta_k}(f(x, y_j)). \tag{8}$$

This head is intentionally not described as an ordinal, error-aware, or preference-strength reward head: it uses none of those labels. It is a handcrafted-feature regression surrogate for constraint satisfaction.

3.7 EXECUTABLE CHECKER AND GATED FUSION

The executable checker applies the parsed constraints directly to each candidate and returns $c_j = c(x, y_j)$. Define candidate-wise standardization

$$\mathrm{z}(v)_j = \frac{v_j - \bar{v}}{\max(\mathrm{sd}(v), \epsilon)} \tag{9}$$

and a residual spread gate

$$\gamma(q) = \min\left(1, \frac{\max_j q_j - \min_j q_j}{\tau}\right). \tag{10}$$

The final score is

$$s_j = \begin{cases} a_j, & g_p(x) = 0, \\ \mathrm{z}(a)_j + \lambda_r \gamma(q)\, \mathrm{z}(q)_j + \lambda_c\, \mathrm{z}(c)_j, & g_p(x) = 1. \end{cases} \tag{11}$$

The prompt gate provides exact locality (Eq. 1). The spread gate addresses a separate problem: when learned predictions are nearly constant, their standardized values can exaggerate numerical noise. It attenuates only the learned residual; the direct checker remains active.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Anchors and benchmark. We use frozen Skywork-Reward-V2 Llama-3.1-8B and Qwen3-8B anchors (Liu et al., 2025a). RewardBench 2 contains 1,865 rows in six equally weighted domains: Factuality, Precise Instruction Following, Math, Safety, Focus, and Ties (Malik et al., 2025). We use the official paired `ref:N/tied:N` Ties calculation, not ordinary row accuracy.

Residual training data. The training pool contains 6,660 prompts and 26,640 candidate responses; 5,860 prompts activate the parser. It combines 2,500 real-response overlays, 960 generated examples for additional constraint families, 2,400 synthetic constrained prompts, and 800 unconstrained controls. Each head uses a 90/10 split by prompt group, 40 epochs, AdamW with learning rate 2×10^{-3} and weight decay 10^{-4}, and batch size 256. The three primary heads use seeds 0, 1, and 2.

Two result tiers. We report a *fixed* configuration, $(\lambda_r, \tau, \lambda_c) = (1.5, 0.5, 2.0)$, that predates the commit-pinned replay. We also report *analysis-selected* configurations from a small grid on all of RewardBench 2. The latter answer mechanistic questions but are not untouched-test estimates. Grouped out-of-fold selection on RewardBench 2 is likewise benchmark-internal because each held-out fold uses labels from the other folds. We do not call CV split seeds independent training seeds.

Paired uncertainty. For the fixed configuration, we compare each Precise-IF decision with the same anchor decision. We report fixes (wrong $\rightarrow$ correct), harms (correct $\rightarrow$ wrong), an exact two-sided McNemar test over discordant items, and a 100,000-replicate paired bootstrap of the six-domain overall difference. The interval is conditional on the saved predictions; it does not include training or model-selection variance.

Reproduction. The replay freezes source commit `77b58385ed7a0df6a2f4591b7efbe7bfba4e8fa8`, hashes both 1,865-row anchor caches, all three checkpoints, and the training pool, and stores candidate-level anchor, residual, checker, and final scores. Row identity uses `(subset,id)` because 40 bare IDs are duplicated across subsets.

4.2 HEADLINE RESULTS

The fixed system improves the reproduced anchors by $+3.23$ pp on Llama and $+3.54$ pp on Qwen. Relative to the published Skywork numbers, the corresponding differences are $+1.34$ pp and $+3.18$ pp. Auto-Rubric reports 0.8091 on Qwen3-8B (Xie et al., 2025); the fixed system is $+0.47$ pp

Table 1: **Official RewardBench 2 results.** "Analysis" selects fusion weights on the full benchmark and is not an untouched-test estimate. Deltas in the text use the exactly reproduced frozen anchors. Published Skywork numbers are shown as external references, not substituted for our anchor caches.

	Llama-3.1-8B		Qwen3-8B	
System	Overall	Precise-IF	Overall	Precise-IF
Published Skywork-V2	0.8410	0.663	0.7820	0.500
Reproduced frozen anchor	0.8221	0.600	0.7783	0.481
CountResidual only (analysis)	0.8398	0.706	0.8002	0.613
Executable checker only (analysis)	0.8523	0.781	**0.8148**	**0.700**
SPECGATE fixed	**0.8544**	**0.794**	0.8138	0.694
SPECGATE best analysis	**0.8544**	**0.794**	**0.8148**	**0.700**

Table 2: **Official domain accuracy for the fixed configuration.** Only Precise-IF changes, which is the metric signature predicted by Eq. 4.

Backbone	System	Fact.	Precise-IF	Math	Safety	Focus	Ties
Llama	Anchor	0.836	0.600	0.765	0.956	0.982	0.794
	SPECGATE	0.836	**0.794**	0.765	0.956	0.982	0.794
Qwen	Anchor	0.802	0.481	0.760	0.940	0.966	0.721
	SPECGATE	0.802	**0.694**	0.760	0.940	0.966	0.721

above it. The analysis-selected Qwen row reaches $+3.65\,\mathrm{pp}$ over our anchor and $+0.57\,\mathrm{pp}$ over Auto-Rubric, subject to the explicit benchmark-selection caveat.

The fixed configuration is slightly better scientifically than the maximized row suggests. It reaches the Llama optimum and gives 0.8138 on Qwen without replay-time parameter selection. On Qwen, the analysis optimum sets $\lambda_r = 0$: the checker is sufficient and the learned residual does not improve the final ordering. On Llama, CountResidual adds $0.21\,\mathrm{pp}$ after the checker.

4.3 THE GAIN IS LOCALIZED EXACTLY WHERE INTENDED

Precise-IF rises by $19.38\,\mathrm{pp}$ on Llama and $21.25\,\mathrm{pp}$ on Qwen under the fixed protocol. The analysis-selected Qwen configuration reaches $+21.88\,\mathrm{pp}$. This clean localization follows from the prompt gate, not from a learned claim of general competence. The parser activates on 167/1,865 prompts: it covers 159/160 Precise-IF prompts (99.4% recall), 95.2% of all activations fall in that domain, and only 8/1,705 non-IF prompts activate (0.47%). All five non-IF domain aggregates remain unchanged under the fixed configuration.

The transition view is more informative than the aggregate alone. Llama corrects eleven times as many cases as it harms; Qwen corrects 9.5 times as many. The net 31 and 34 items imply $31/(160\times6) = 3.229\,\mathrm{pp}$ and $34/(160\times6) = 3.542\,\mathrm{pp}$ overall, exactly matching the observed changes. Paired 95% bootstrap intervals for the overall gain are $[2.08, 4.38]\,\mathrm{pp}$ and $[2.40, 4.79]\,\mathrm{pp}$. The lower bounds remain well above zero even though these intervals deliberately exclude training uncertainty.

The fixes are broad; the harms are concentrated. Llama's 34 fixes span 21 parsed constraint families and Qwen's 38 fixes span 19; no family contributes more than four fixes on either backbone. This rules out a single high-frequency template as the headline explanation. In contrast, four of the seven total harm events occur on `emoji_each_sentence`; the remaining three involve Python-only output, sentence-type balance, and palindromes. The concentration points to sentence segmentation and exact boundary semantics as the next checker improvements, rather than to a need for a larger residual model.

Table 3: **Paired Precise-IF transitions for the fixed configuration.** "Fix" and "harm" compare the same item before and after reranking.

Backbone	Anchor correct	Final correct	Fix	Harm	Net	McNemar p
Llama-3.1-8B	96/160	127/160	34	3	$+31$	1.23×10^{-7}
Qwen3-8B	77/160	111/160	38	4	$+34$	5.65×10^{-8}

Table 4: **First-principles component decomposition** under official scoring. Each component row is analysis-selected, so it diagnoses contribution rather than estimating held-out performance.

System	Llama	Δ anchor	Qwen	Δ anchor
Frozen anchor	0.8221	—	0.7783	—
CountResidual only	0.8398	$+1.77$ pp	0.8002	$+2.19$ pp
Checker only	0.8523	$+3.02$ pp	**0.8148**	$+\mathbf{3.65}$ pp
Full system	**0.8544**	$+\mathbf{3.23}$ pp	**0.8148**	$+\mathbf{3.65}$ pp

4.4 Where the gain actually comes from

Before interaction effects, the checker accounts for approximately 94% of the best Llama gain and 100% of the best Qwen gain. In item counts, CountResidual alone fixes a net 17 Llama and 21 Qwen Precise-IF decisions; the checker alone fixes 29 and 35. Adding the residual after the checker fixes two additional Llama items, but the fixed Qwen residual loses one of the checker's corrections and the analysis optimum sets $\lambda_r = 0$.

This asymmetry explains the role of learning. CountResidual is useful when soft, distributed count evidence resolves cases left near the anchor–checker boundary; it is not useful merely because it has trainable parameters. The checker already exposes the sufficient statistic for Qwen's remaining errors. The strongest empirical lesson is therefore not "learn a larger correction," but "learn only what the executable path does not already settle."

4.5 Both gates are necessary

The prompt gate is the main safety mechanism. Without it, the Precise-IF score remains high, but unrelated domains pay for the intervention. Relative to the anchor, ungated Llama loses $2.95/5.46/4.22/4.85$ pp on Factuality/Math/Safety/Focus; ungated Qwen loses $2.32/4.92/3.11/5.86$ pp. These losses almost cancel the intended IF gain, exactly as Eq. 2 predicts. The spread gate addresses the separate scale paradox in Eq. 3: removing it costs 1.18 pp on Llama and 0.73 pp on Qwen even after retuning the remaining weights.

4.6 What CountResidual learns

Prompt specification and raw response counts carry most of the transferable learned signal. Explicit interaction features are largely redundant because the MLP can combine requested values and observed counts itself (Appendix Table 9). Dropping the specification costs $1.18/1.29$ pp on Llama/Qwen; dropping raw counts costs $0.73/0.63$ pp. No single constraint family changes the fixed full-system score by more than 0.42 pp, which argues against one isolated pattern explaining the result.

Real responses matter. Training CountResidual only on toy synthetic templates produces 0.8020/0.7672 on Llama/Qwen, below both frozen anchors. Using overlays on real responses and the new-family examples reaches 0.8377/0.7992, close to the full pool's 0.8398/0.8002. The useful transfer comes from preserving realistic response texture while varying literal compliance, not from generating more templated examples.

Table 5: **Gate ablation.** Best official score within each ablated analysis grid.

Configuration	Llama	Qwen
Prompt gate + spread gate	**0.8544**	**0.8148**
No spread gate	0.8426	0.8075
No prompt gate	0.8226	0.7818
Neither gate	0.5402	0.5085

5 Limitations and Future Work

Benchmark selection. The analysis-best fusion weights and feature-ablation weights are selected on RewardBench 2 itself. Grouped out-of-fold analysis reduces direct per-row reuse but still uses labels from the same benchmark. A new benchmark or prospectively frozen protocol is required for a clean generalization claim.

Parser coverage and brittleness. The parser covers literal, countable instructions. It can miss paraphrases, misparse ambiguous requirements, or incorrectly treat a semantic request as mechanical. Exact locality prevents damage when it abstains; it does not prevent damage after a false-positive parse. Adversarial paraphrase and parser-precision evaluation are therefore required.

Reranking, not a global scalar reward. Per-candidate-set standardization makes the final score context dependent. The system is appropriate for best-of-n selection but does not define a global transitive utility over independently scored responses. Removing this dependence without amplifying scale mismatch is an open engineering problem.

Structured residual learning. A separate ongoing direction learns a bounded scalar residual and an evidence-based acceptance gate from aspect, error, and preference-strength labels under worst-tail break constraints. That method is not CountResidual, and its unfinished benchmark results are not mixed into this paper. It is a natural next step for semantic errors that cannot be delegated to an executable checker.

6 Conclusion

SpecGate separates broad semantic judgment from explicit contract verification. Support gating keeps the specialist inside its competence region, while spread gating suppresses weak learned evidence. On two 8B anchors, the fixed system gains $3.23/3.54\,\mathrm{pp}$ through 31/34 net Precise-IF corrections without moving the other five domains. The checker supplies most of the gain; the residual adds two Llama corrections and is unnecessary at the Qwen optimum. The resulting principle is: *localize first, verify second, and learn only the residual that remains*.

AI Use Statement

Generative AI tools were used to check our paper.

Ethics Statement

This work studies reranking of already generated text and introduces no human-subject data collection. The executable verifier is intentionally limited to explicit, mechanically checkable constraints. It should not be used as a substitute for semantic safety, factuality, or fairness evaluation. The authors reviewed the ICLR Code of Ethics and remain responsible for all claims and released artifacts.

Reproducibility Statement

Section 4.1 specifies the fixed and analysis-only protocols; Sections 4 and Appendix E separate primary measurements from mechanism analyses and archival records. Appendices B–E provide

evaluator details, complete hyperparameters, data composition, statistical procedures, and provenance. The supplement includes a script that recomputes all paired tests from candidate-level outputs.

REFERENCES

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022. doi: 10.48550/arXiv.2204.05862. URL https://arxiv.org/abs/2204.05862.

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. doi: 10.2307/2334029.

Paul F. Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/d5e2c0adad503c91f91df240d0cd4e49-Paper.pdf.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. doi: 10.48550/arXiv.2110.14168. URL https://arxiv.org/abs/2110.14168.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10835–10866. PMLR, 2023. URL https://proceedings.mlr.press/v202/gao23h.html.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/4a8423d5e91fda00bb7e46540e2b0cf1-Paper.pdf.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Sx038qxjek.

Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. FollowBench: A multi-level fine-grained constraints following benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 4667–4688. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.257. URL https://aclanthology.org/2024.acl-long.257/.

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*, pp. 1755–1797. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-naacl.96. URL https://aclanthology.org/2025.findings-naacl.96/.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison 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*, pp. 39578–39601, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/aca97732e30bcf1303bc22ac3924fd16-Paper-Conference.pdf.

Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, Yang Liu, and Yahui Zhou. Skywork-reward-V2: Scaling preference data curation via human–ai synergy. *arXiv preprint arXiv:2507.01352*, 2025a. doi: 10.48550/arXiv.2507.01352. URL https://arxiv.org/abs/2507.01352.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. RM-Bench: Benchmarking reward models of language models with subtlety and style. In *International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=QEHrmQPBdd.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46534–46594. Curran Associates, Inc., 2023. doi: 10.52202/075280-2019. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/91edff07232fb1b55a505a9e9f6c0ff3-Paper-Conference.pdf.

David Madras, Toni Pitassi, and Richard Zemel. Predict responsibly: Improving fairness and accuracy by learning to defer. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/09d37c08f7b129e96277388757530c72-Paper.pdf.

Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Morrison, Noah A. Smith, Hannaneh Hajishirzi, and Nathan Lambert. RewardBench 2: Advancing reward model evaluation. *arXiv preprint arXiv:2506.01937*, 2025. doi: 10.48550/arXiv.2506.01937. URL https://arxiv.org/abs/2506.01937.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. doi: 10.52202/068431-2011. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. InFoBench: Evaluating instruction following ability in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13025–13048. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.772. URL https://aclanthology.org/2024.findings-acl.772/.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, pp. 53728–53741. Curran Associates, Inc., 2023. doi: 10.52202/075280-2338. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/a85b405ed65c6477a4fe8302b5e06ce7-Paper-Conference.pdf.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021. Curran Associates, Inc., 2020. URL https://arxiv.org/abs/2009.01325.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022. doi: 10.48550/arXiv.2211.14275. URL https://arxiv.org/abs/2211.14275.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. HelpSteer 2: Open-source dataset for training top-performing reward models. In *Advances in Neural Information Processing*

Systems, volume 37, pp. 1474–1501. Curran Associates, Inc., 2024. doi: 10.52202/079017-0047. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/02fd91a387a6a5a5751e81b58a75af90-Paper-Datasets_and_Benchmarks_Track.pdf.

Zijun Weng, Xiaohui Hu, Shuangyong Song, Yongxiang Li, Kaidong Yu, and Xuanjing Huang. Prompt-level reward specifications for open-ended post-training. *arXiv preprint arXiv:2605.29275*, 2026. doi: 10.48550/arXiv.2605.29275. URL https://arxiv.org/abs/2605.29275.

Lipeng Xie, Sen Huang, Zhuo Zhang, Anni Zou, Yunpeng Zhai, Dingchao Ren, Kezun Zhang, Haoyuan Hu, Boyin Liu, Haoran Chen, Zhaoyang Liu, and Bolin Ding. Auto-rubric: Learning from implicit weights to explicit rubrics for reward modeling. *arXiv preprint arXiv:2510.17314*, 2025. doi: 10.48550/arXiv.2510.17314. URL https://arxiv.org/abs/2510.17314.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023. doi: 10.48550/arXiv.2306.05685. URL https://arxiv.org/abs/2306.05685.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023. doi: 10.48550/arXiv.2311.07911. URL https://arxiv.org/abs/2311.07911.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019. doi: 10.48550/arXiv.1909.08593. URL https://arxiv.org/abs/1909.08593.

SUPPLEMENTARY MATERIAL

This supplement contains complete derivations, evaluator and statistical details, implementation specifications, additional mechanism analyses, and the provenance boundary for historical results. No claim in the main paper depends on reading the supplement.

A COMPLETE DERIVATIONS

A.1 SUPPORT-MISMATCH DECOMPOSITION

Let $I(x) = \mathbf{1}[x \in \mathcal{V}]$ and let $\Delta U(x) = U(s; x) - U(a; x)$. By the law of total expectation,

$$\mathbb{E}[\Delta U] = \Pr(I = 1)\mathbb{E}[\Delta U \mid I = 1] + \Pr(I = 0)\mathbb{E}[\Delta U \mid I = 0] \tag{12}$$

$$= \rho G_{\mathcal{V}} + (1 - \rho)G_{\neg\mathcal{V}}. \tag{13}$$

If $G_{\neg\mathcal{V}} = -\epsilon$ for $\epsilon > 0$, positive total gain requires $G_{\mathcal{V}} > (1 - \rho)\epsilon/\rho$. The empirical activation rate is $\rho = 167/1{,}865 = 0.0895$, yielding $(1 - \rho)/\rho = 10.17$. This calculation concerns per-prompt risk and is distinct from RewardBench 2's six-domain macro-average.

A.2 PROOF OF SUPPORT SAFETY

For every $x \notin \mathcal{V}$, Eq. 11 selects its first branch and returns $s_j = a_j$ for every candidate j. Therefore

$$\operatorname{sign}(s_j - s_k) = \operatorname{sign}(a_j - a_k)$$

for every candidate pair (j, k). The reranker introduces neither a new strict preference nor a new tie outside its declared support.

A.3 PROOF OF VANISHING-EVIDENCE CONTINUITY

For a nonconstant vector $q \in \mathbb{R}^n$, standardization gives zero mean and unit population variance. Hence each coordinate satisfies $|\,\mathrm{z}(q)_j| \le \sqrt{n-1}$. The numerical standard-deviation floor can only reduce this magnitude. Therefore

$$\|\gamma(q)\,\mathrm{z}(q)\|_\infty \le \sqrt{n-1}\,\min\left(1, \frac{\operatorname{range}(q)}{\tau}\right),$$

which converges to zero with $\operatorname{range}(q)$. Without γ, standardization is scale-invariant above the numerical floor and does not provide this operational attenuation.

A.4 MACRO-SCORE ACCOUNTING

Let A_d be the score of domain d. The official RewardBench 2 overall is $B = \frac{1}{6}\sum_{d=1}^{6} A_d$. Under the fixed protocol, the five non-IF domain scores are identical before and after reranking. Therefore

$$\Delta B = \frac{\Delta A_{\mathrm{IF}}}{6}.$$

For Llama, 31 net corrections among 160 IF items give $\Delta A_{\mathrm{IF}} = 31/160 = 0.19375$ and $\Delta B = 31/(160 \cdot 6) = 0.0322917$. For Qwen, 34 net corrections give $34/(160 \cdot 6) = 0.0354167$. These equal the measured overall changes.

B EVALUATOR AND STATISTICAL DETAILS

B.1 OFFICIAL REWARDBENCH 2 AGGREGATION

For Factuality, Precise-IF, Math, Safety, and Focus, an item is correct when every chosen candidate scores above every rejected candidate. The official overall is the unweighted mean of all six domain scores. Ties is not independent row accuracy: 51 `ref:N/tied:N` groups are paired and combined from reference accuracy, tied accuracy, correctness-preferred, hard correctness-preferred, and a bounded margin term using the upstream implementation at commit `e4050493415ed7183a38d1778e148c49904f6f05`.

B.2 PAIRED INFERENCE

For each fixed model family, we compute binary correctness before and after reranking on the same 160 Precise-IF items. A fix is $(0, 1)$ and a harm is $(1, 0)$. The exact two-sided McNemar test is the conditional binomial test over the discordant count. The confidence interval resamples the 160 paired differences 100,000 times with seed 20260925 and divides each sampled IF mean by six to express the corresponding overall-score change. All 160 prompt hashes are unique.

Table 6: Paired inference for the fixed configuration. Confidence intervals are conditional on saved predictions and exclude training variance.

Backbone	Net fixes	Overall Δ	Paired 95% CI	Exact p
Llama-3.1-8B	31	$3.229\,\text{pp}$	$[2.083, 4.375]\,\text{pp}$	1.23×10^{-7}
Qwen3-8B	34	$3.542\,\text{pp}$	$[2.396, 4.792]\,\text{pp}$	5.65×10^{-8}

C IMPLEMENTATION DETAILS

Table 7: Locked implementation and training configuration.

Component	Value
Constraint families	38 including `other`
Specification / count / interaction / extra dimensions	44 / 18 / 10 / 8
CountResidual MLP	80–64–64–1, GELU
Parameters per head / ensemble	9,409 / 28,227
Heads / training seeds	3 / 0, 1, 2
Optimizer / epochs / batch size	AdamW / 40 / 256
Learning rate / weight decay	2×10^{-3} / 10^{-4}
Train/validation split	90/10 by prompt group
Fixed $(\lambda_r, \tau, \lambda_c)$	$(1.5, 0.5, 2.0)$
Official evaluator	`rewardbench2-official-e405049-v1`

Feature ordering. The 18 response statistics are coordinating conjunctions, numbers, pronouns, words, person names, newlines, parenthesis depth, quote depth, palindromes, punctuation coverage, interrobang presence, sentence-final emoji ratio, one-word-line ratio, indentation monotonicity, HTML italics, list markers, `!?!?` markers, and unique words. Ten interaction features cover phrase prefix/suffix/inclusion and normalized or absolute count-to-target relations. Eight extra features cover keyword sentence position, sentence-type balance, declarative/interrogative ratio, word-count steps, consecutive initials, distinct vowels, increasing alliteration, and short multiple-choice answers.

C.1 RESIDUAL TRAINING POOL

Table 8: Composition of the CountResidual training pool. Every row has four candidate responses.

Source	Prompts	Responses	Purpose
Real-response overlays	2,500	10,000	Realistic text with varied compliance
Additional constraint families	960	3,840	Coverage beyond base templates
Synthetic constrained prompts	2,400	9,600	Controlled satisfaction levels
Unconstrained controls	800	3,200	Learn neutral behavior
Total	6,660	26,640	5,860 parser-positive prompts

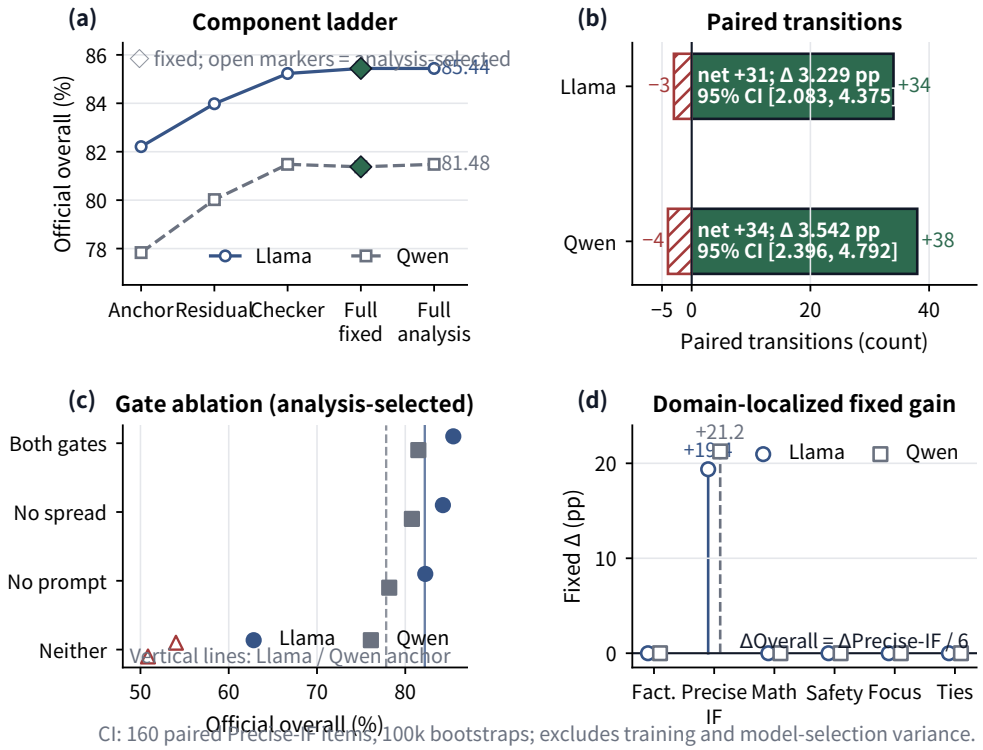

Figure 3: **Mechanism analysis.** Panels show (a) the component ladder, (b) fix/harm transitions with paired intervals, (c) prompt- and spread-gate ablations, and (d) the domain delta profile. Fixed-protocol results are distinguished from benchmark-internal analysis.

Each head is split by complete prompt groups before candidates are flattened. The final layer is zero-initialized, and constrained responses receive weight 3 while unconstrained controls receive weight 1 in the MSE objective.

C.2 Matched feature ablation

Table 9: Residual-only feature ablation. Three heads are retrained with one block zeroed; scores use official aggregation but analysis-selected fusion weights.

Residual features	Llama	Change	Qwen	Change
All 80 features	**0.8398**	—	**0.8002**	—
Drop prompt specification	0.8280	−1.18 pp	0.7874	−1.29 pp
Drop response counts	0.8325	−0.73 pp	0.7940	−0.63 pp
Drop explicit interactions	0.8402	+0.04 pp	0.7992	−0.10 pp
Drop extra-family proxies	0.8356	−0.42 pp	0.7992	−0.10 pp

D Additional Mechanism Analysis

D.1 Constraint-family distribution

The fixed Llama system fixes cases in 21 parsed families; the fixed Qwen system fixes cases in 19. No family contributes more than four fixes. Harms are much more concentrated: four of seven total harm events are in `emoji_each_sentence`. The remaining harms occur in `python_only`,

`sentence_type_balance`, and `palindromes`. This pattern motivates a dedicated sentence-segmentation and boundary-semantics audit.

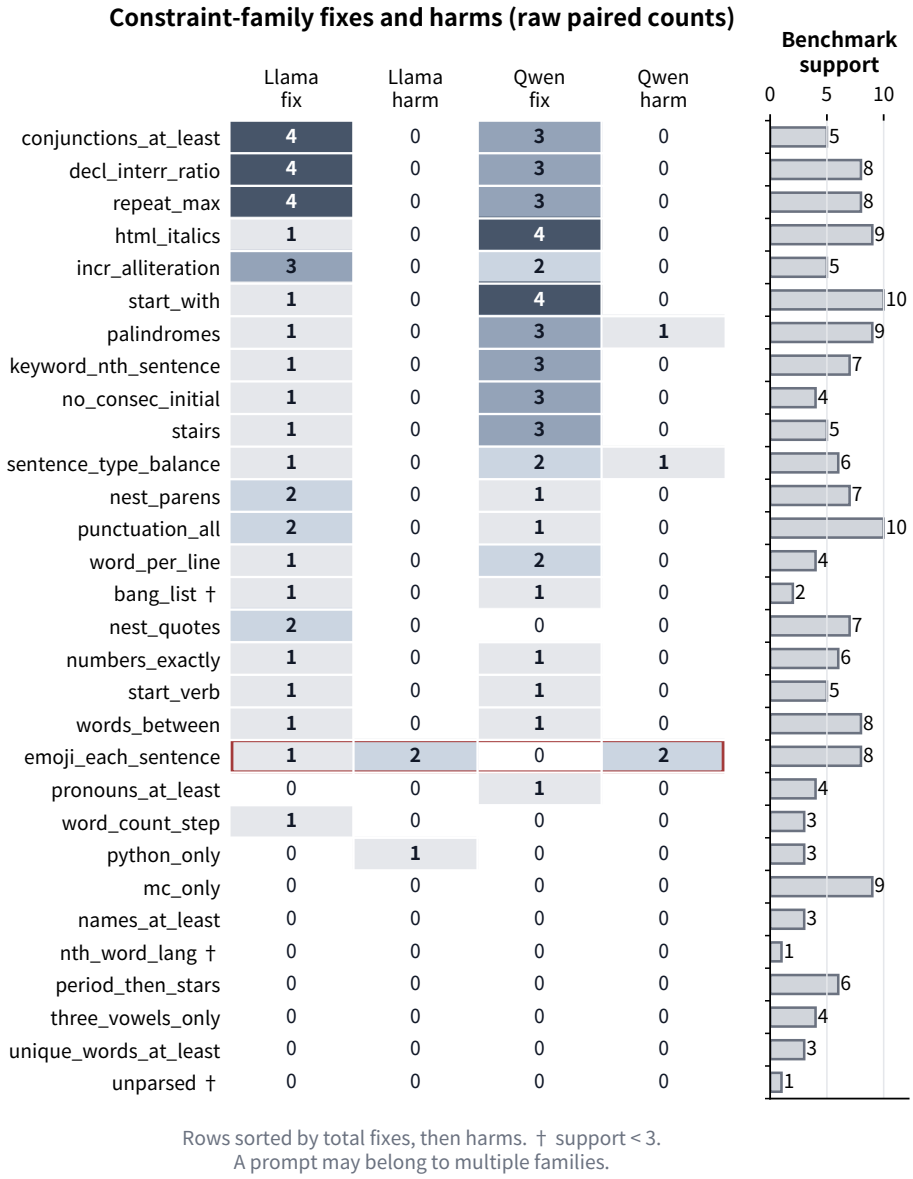

Figure 4: Planned supplementary family analysis: fixes and harms by parsed constraint family for both anchors. The final asset must show counts, not normalized percentages, and mark families with fewer than three observations.

D.2 Fusion sensitivity

The fixed point is $(\lambda_r, \tau, \lambda_c) = (1.5, 0.5, 2.0)$. The analysis grid contains $\lambda_r \in \{0, 1, 1.5, 2\}$, $\tau \in \{0.4, 0.5, 0.6\}$, and $\lambda_c \in \{0, 1.5, 2, 2.5\}$. The Qwen optimum includes $\lambda_r = 0$, while several Llama configurations tie at the optimum. The surface should therefore be reported as a plateau rather than as evidence for one precisely identified hyperparameter.

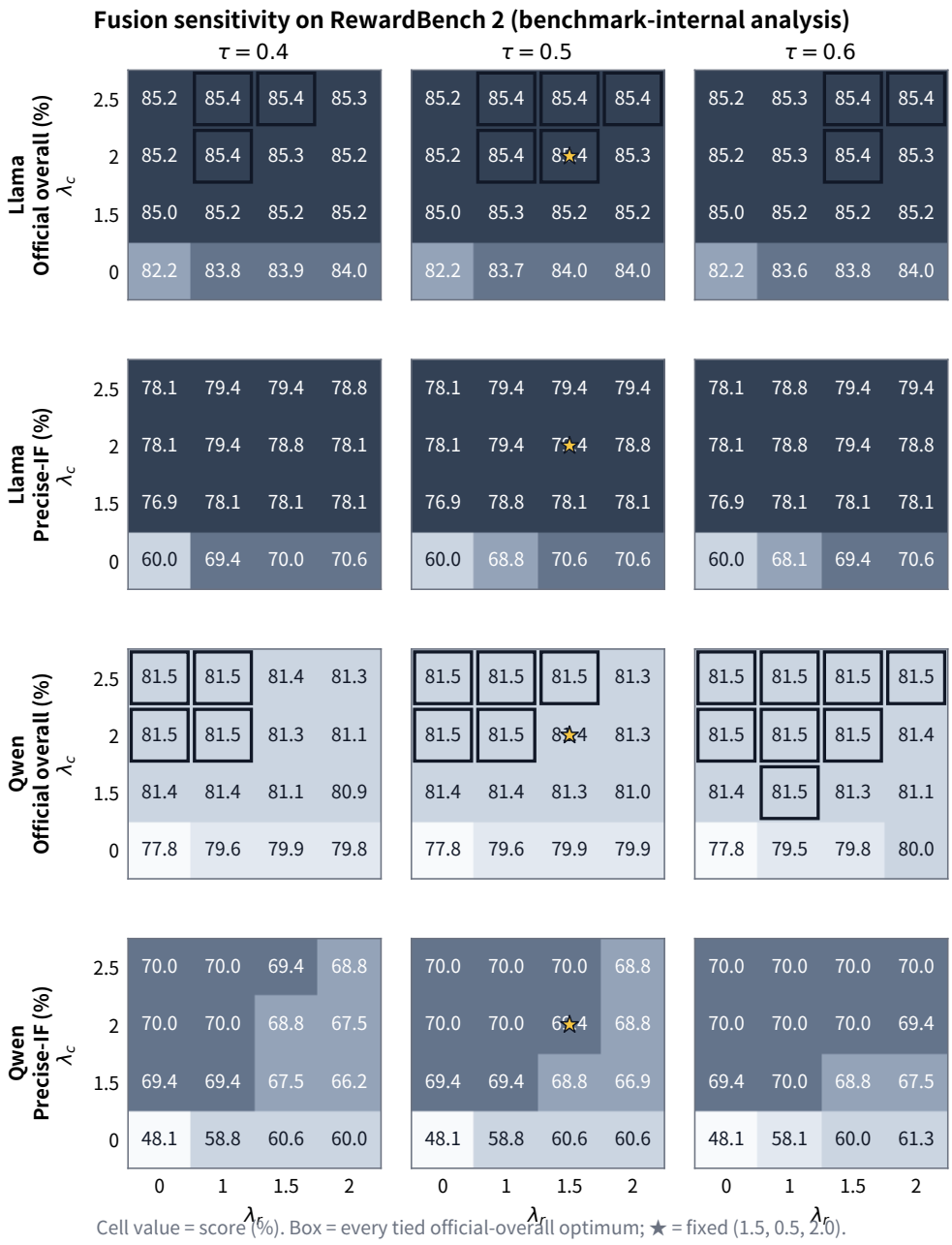

Figure 5: Planned supplementary sensitivity visualization: official overall and Precise-IF over the fusion grid, with the fixed configuration and all tied optima marked explicitly.

E Historical Results and Provenance Boundaries

E.1 Historical metric and official correction

The absolute correction comes entirely from replacing row-wise Ties accuracy with the official paired Ties metric. Because SpecGate leaves all Ties scores unchanged, the anchor-relative gains are identical under both evaluators.

Table 10: Recorded headline, exact replay under the historical metric, and official rescoring.

Backbone	Archived record	Historical-metric replay	Official fixed
Llama-3.1-8B	0.8721	0.8710	0.8544
Qwen3-8B	0.8268	0.8268	0.8138

Evaluator-corrected reproduction. Replaying the archived implementation produces 87.10 on Llama and 82.68 on Qwen under its original metric; the remaining difference from the recorded 87.21 Llama score is one Precise-IF item. The historical evaluator treated the 102 Ties rows as independent examples, whereas the official metric pairs 51 reference/tied groups and evaluates both ranking correctness and tie behavior. Replacing only that evaluator logic gives the official fixed scores in Table 10.

The replay also corrects the scientific identity of the archived result. The checkpoints are CountResidual MLPs over handcrafted features, not ordinal reward heads, and the full system includes a live executable checker. Human aspect ratings, error tags, and preference strength are not used.

E.2 Claim boundaries

The evidence establishes that prompt-gated constraint verification can correct a concentrated Reward-Bench 2 failure mode. It does not establish a general improvement in reward modeling, robustness to semantic constraints, or an independent held-out benchmark gain. It also does not establish that the learned residual is necessary on every anchor: Qwen is a direct counterexample.

E.3 Archived size scaling

The following values motivated this study but were not all reconstructed with candidate-level provenance or official Ties scoring. They are retained as archival records, not main-paper evidence.

Table 11: Archive-only size record; this is not an officially revalidated scaling law.

Backbone	Published	Internal base	Archived system	Δ published
Qwen3-0.6B	0.613	0.6247	0.6684	$+5.54$ pp
Qwen3-1.7B	0.683	0.6922	0.7349	$+5.19$ pp
Qwen3-4B	0.755	0.7666	0.8083	$+5.33$ pp
Qwen3-8B	0.782	0.7921	0.8268	$+4.48$ pp
Llama-3.1-8B	0.841	0.8392	0.8721	$+3.11$ pp