ANNOTATE OR FILTER? THE REDUNDANCY–RESCUE MECHANISM AND PHASE BOUNDARIES IN LLM FUSION PIPELINES

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-component pipelines such as Archon place a verifier in front of a fuser, assuming that discarding incorrect candidates can only help synthesis. We show that this selection-before-synthesis prior is mechanistically inverted. On MATH-500, five 8B process reward models all degrade fusion: verifier-filtered pipelines reach 0.818–0.866 accuracy versus 0.888 for unfiltered critique-and-fuse, a loss of 2.2–7.0 points. The cause is not verifier incompetence but a redundancy–rescue mechanism. Generator outputs are strongly polarized—most problems are either entirely correct or entirely wrong—so a verifier can act only in the thin mixed tier, where false positives pass errors through while false negatives are absorbed by redundancy (80.2% of problems have at least two correct candidates). The fuser, conversely, is an error corrector: it rescues 41.0% of majority-vote failures and corrupts 1.4%, synthesizing correct answers even from all-wrong sets and thereby breaking the generator ceiling (0.834 $\rightarrow$ 0.888). Filtering therefore starves fusion of the incorrect candidates that constitute its corrective signal. We derive a three-tier accuracy decomposition that predicts both the FP/FN asymmetry and a filtering/fusion phase boundary from two observable statistics—candidate redundancy and fuser rescue rate. Strong fusors should receive annotated but unfiltered candidates; weak fusors still benefit from gating. The crossover lies near FP $\approx$ 0.25, well below the operating regime of current 8B PRMs (0.41–0.64). Dual-path cross-validation against a 60-point FP/FN grid and a weak-fuser replication confirm both. The implication is architectural: verifiers should annotate, not filter.

1 INTRODUCTION

Multi-component inference pipelines increasingly structure LLM applications as chains of specialized models: a generator produces candidate solutions, a verifier scores them, and a fuser—an LLM that synthesizes one answer from multiple candidates and critiques (Jiang et al., 2023; Wang et al., 2024a)—produces the final output (Saad-Falcon et al., 2024; Wang et al., 2022; Snell et al., 2024; Wei et al., 2022; Yao et al., 2023; Brown et al., 2024). The dominant architectural assumption is **selection-before-synthesis**: filter out incorrect candidates with a verifier, then fuse only the survivors. The intuition is straightforward—why feed known-bad solutions to a downstream model?

We show that this intuition is mechanistically inverted for strong fusion models. On MATH-500, five different 8B process reward models (PRMs) all *degrade* fusion accuracy: verifier-filtered pipelines reach 0.818–0.866 versus 0.888 for unfiltered fusion, a loss of 2.2–7.0 percentage points. The cause is not verifier incompetence but a **redundancy–rescue mechanism**: generator outputs are strongly polarized (70.2% all-correct, 16.6% all-wrong, only 13.2% mixed), so verifiers can only act in a thin tier where false positives pass errors through while false negatives are absorbed by redundancy. The fuser, critically, is an error corrector—unlike standalone self-correction where models struggle to identify their own errors without external signal (Huang et al., 2023; Madaan et al., 2023; Shinn et al., 2023)—it rescues 41.0% of majority-vote failures by recombining valid fragments from incorrect candidates. Filtering therefore starves fusion of the very material that constitutes its corrective signal.

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.

We derive an analytical framework that predicts both the FP/FN asymmetry and a filtering/fusion **phase boundary** from two observable statistics: candidate redundancy (ρ) and fuser rescue rate (α_R). The framework yields a simple decision rule: strong fusors should receive annotated but unfiltered candidates (critique as annotation, not as a gate); weak fusors still benefit from gating. The empirical crossover lies near $\overline{\text{FP}} \approx 0.25$, well below the operating regime of current 8B PRMs ($\overline{\text{FP}} \in [0.41, 0.64]$). We translate this rule into **CaA-Protocol**, a system-level configuration protocol that determines, from lightweight offline calibration, whether a pipeline should annotate or filter.

Our contributions are threefold:

- **C1 – Redundancy–Rescue Mechanism (phenomenon).** We identify and empirically validate a mechanistic inversion in fusion pipelines: verifier filtering harms strong fusors because incorrect candidates contain the corrective ingredients that fusors recombine. We provide dual-path evidence (five real 8B PRMs and a 60-point simulated grid), step-level micro-evidence of ingredient recycling from rejected candidates, and a stratified accuracy decomposition ($390 + 34 + 26 - 6 = 444$, $444/500 = 0.888$) that breaks the generator-side selection ceiling (0.834).
- **C2 – Analytical Framework (theory).** We formalize the redundancy–rescue mechanism as a Signal–Noise Trade-off framework with Suppressed Fusion Formulation: CaA (full ingredients) vs. Gate (filtered ingredients suppressing effective rescue rate α_R). The unified expected-gain equation $\Delta\text{Acc} = [\alpha_R - \alpha_R(\text{Gate})] \cdot P(G_5 \text{ wrong}) - \beta_{\text{Fuser}} \cdot (1 - \rho)$ explains both the strong-fuser regime (Plus: CaA wins) and the weak-fuser regime (Turbo: Gate wins), and yields a phase boundary at $\overline{\text{FP}} \approx 0.25$.
- **C3 – CaA-Protocol (method).** We operationalize the phase boundary as a system-level configuration protocol with offline parameter profiling (no per-query ground truth required). The protocol determines whether a given generator–verifier–fuser triplet should deploy annotation or filtering, validated on two regimes spanning a 15.2pp accuracy gap between the strongest and weakest fuser configurations.

2 Experimental Setup: Dual-Path Audit

We design a dual-path audit to disentangle verifier quality from fusion capability. Path A measures real 8B PRMs on every problem; Path B simulates verifiers across a 60-point FP/FN grid to isolate each error type's effect.

2.1 Architecture Components

A fusion pipeline consists of three components:

Generator G. Produces $K = 5$ candidate solutions per problem via temperature sampling ($T = 0.7$). We use Qwen-Plus as the primary generator and Qwen-Turbo for the weak-fuser ablation.

Verifier V. Assigns a scalar score to each candidate. We evaluate five open-weight 8B PRMs: Skywork-o1-8B-PRM (AI, 2024), EurusPRM-Stage1 and EurusPRM-Stage2 (Yuan et al., 2025), Qwen2.5-Math-PRM-7B (Team, 2024b), and GPM-8B (Team, 2024a), a set that spans the dominant step-level process-supervision paradigms (Lightman et al., 2023; Wang et al., 2024b; Uesato et al., 2022; Shao et al., 2024; Luo et al., 2023). For Path B we simulate verifiers at $\overline{\text{FP}}, \overline{\text{FN}} \in \{0.0, 0.05, 0.1, 0.15, 0.2, 0.3\}$ (10 FP $\times$ 6 FN = 60 grid points) by flipping correctness labels independently per candidate.

Fuser F. An LLM that synthesizes one answer from multiple candidates and optionally from per-candidate critiques. We use Qwen-Plus (strong fuser, primary) and Qwen-Turbo (weak fuser, ablation). Critiques are generated by prompting the same model used as generator to produce step-by-step commentary on each candidate.

2.2 Pipeline Architectures

We define five architectures by composing G, V, and F:

- G_1: Single best-of-1 candidate.

- G_5: Majority vote over $K = 5$ candidates.
- G_5+V: Select the top-scoring candidate by verifier V (no fusion).
- G_5+V+F (Gate – Filtered Fusion): Filter candidates by PRM $\geq \tau$, pass survivors to F; fall back to top-1 if all filtered. Thresholds: $\tau = 0.5$ for GPM, QwenPRM, EurusPRM-S2; $\tau = 0.1$ for Skywork, EurusPRM-S1.
- $G_5 + C + F$ (CaA – Critique-as-Annotation): Pass all K candidates with per-candidate critiques to F without filtering. This is our proposed method.

The perfect verifier V^* (oracle, $\overline{\text{FP}}=\overline{\text{FN}}=0$) yields G_5+V^*, the pure selection upper bound.

2.3 Dual-Path Cross-Validation

Path A (Real Verifiers). We run all five 8B PRMs on the full MATH-500, measuring $\overline{\text{FP}}$, $\overline{\text{FN}}$, and end-to-end accuracy for G_5+V and G_5+V+F. Cross-validation: for 4/5 verifiers, $|G_5+V+F - \text{CaA}| \leq 0.04$; GPM is an outlier ($\Delta = -0.072$) whose $\overline{\text{FP}} = 0.635$ exceeds the simulated grid range, consistent with the phase boundary prediction.

Path B (Simulated Grid). We simulate verifiers at every FP/FN combination by randomly flipping candidate correctness labels. This isolates the effect of each error type independent of verifier-specific biases and covers the full FP/FN space including regions ($\overline{\text{FP}} < 0.25$) where no real 8B PRM operates.

2.4 Data and Metrics

MATH-500 (Lightman et al., 2023; Hendrycks et al., 2021) ($N = 500$): competition-level mathematics with step-by-step solutions. Answer extraction uses a three-tier protocol: last occurrence of "answer is:" $\rightarrow$ `\boxed{}` $\rightarrow$ `$$...$$` with nested-brace balancing and sympy equivalence scoring. **GSM8K** (Cobbe et al., 2021) ($N = 50$): grade-school math used only as a boundary condition (Section 6).

Metrics. All accuracies are exact-match after LaTeX normalization. Statistical significance for rescue/corruption rates uses Wilson binomial 95% confidence intervals. Verifier FP/FN are computed per-problem against ground-truth candidate correctness (n_{correct} labels). The fuser rescue rate α_R is defined as $\Delta_{\text{all-wrong}}/N_{\text{all-wrong}}$ unless otherwise specified; the corruption rate α_C is $\Delta_{\text{corruption}}/N_{G_5 \text{ correct}}$.

2.5 Reproducibility

All experiments use API inference (DashScope Qwen-Plus/Turbo) with fixed hyperparameters ($T = 0.7$, max_tokens: MATH=4096, AIME=9192, GSM8K=1024). Verifier scoring is offline (8B PRMs run once; Path B grid run once). No per-problem inference was repeated. Full configuration, prompt templates, and answer extraction code will be released upon acceptance. The canonical result file is `plus_architecture_results_fixed.json` together with Path A (`rq1_3_path_a_results.json`) and Path B (`rq1_3_path_b_extended_results.json`) JSON outputs. All key numerical claims in this paper can be traced to these three files.

3 Analytical Framework: Signal–Noise Trade-off and Phase Boundaries

This section formalizes the redundancy–rescue mechanism as an analytical framework, derives the phase boundary from observable statistics, and establishes the conditions under which filtering helps or harms fusion.

3.1 Notation and Preliminaries

Let a generator G produce $K = 5$ candidates $\{X_1, \ldots, X_K\}$ per problem. Define per-problem correctness tier by $n_{\text{correct}} \in \{0, \ldots, K\}$:

- **All-correct** ($n_{\text{correct}} = K$): $p_{\text{all}} = 0.702$ (351/500).
- **Mixed** ($0 < n_{\text{correct}} < K$): $p_{\text{mix}} = 0.132$ (66/500).
- **All-wrong** ($n_{\text{correct}} = 0$): $p_{\text{none}} = 0.166$ (83/500).

Candidate redundancy $\rho := P(n_{\text{correct}} \geq 2) = 0.802$ (401/500). A verifier V has empirical $\overline{\text{FP}}$, $\overline{\text{FN}}$; real 8B PRMs operate at $\overline{\text{FP}} \in [0.41, 0.64]$. A fuser F is an LLM synthesizing one answer from candidates and critiques. Rescue capacity $\alpha_R := P(F \text{ correct} \mid \text{majority wrong}) = 0.410$ (34/83 all-wrong); corruption $\alpha_C = 0.014$ (6/390). Two pipelines: **CaA** ($G_5{+}C{+}F$, no filter) and **Gate** ($G_5{+}V{+}F$, PRM-filtered, Top-1 fallback).

3.2 Analysis 1: The Generator-Side Ceiling

By definition, a perfect verifier ($\overline{\text{FP}} = 0$, $\overline{\text{FN}} = 0$) can only select from existing correct candidates. Since it cannot rescue all-wrong problems, its accuracy ceiling is

$$\text{Acc}(G_5{+}V^*) = 1 - p_{\text{none}} = 0.834. \tag{1}$$

This is the **Selection Ceiling**—a pure verifier selection bound without any fuser, analogous to the verifier-side ceiling characterized in prior work (Stroebl et al., 2026; Lightman et al., 2023). We emphasize that this is an algebraic identity, not a theorem: $1 - p_{\text{none}}$ follows directly from the definition of p_{none} as the fraction of problems with zero correct candidates.

3.3 Analysis 2: Fusion Breakthrough—Stratified Accounting

When the fuser can recombine valid fragments from incorrect candidates, it can break the Selection Ceiling. We decompose the observed accuracy gain into a stratified net-gain ledger:

$$\text{Acc}(\text{CaA}) = \text{Acc}(G_5) + \frac{\Delta_{\text{all-wrong}} + \Delta_{\text{mixed}} - \Delta_{\text{corruption}}}{N}. \tag{2}$$

On $N = 500$ MATH problems, G_5 solves 390 ($\text{Acc}(G_5) = 0.780$) and CaA solves 444 ($\text{Acc}(\text{CaA}) = 0.888$), a net gain of $+54$ problems ($+10.8$pp). This decomposes into three strata:

1. **All-Wrong Breakthrough:** In the 83 all-wrong problems, the fuser rescues $\Delta_{\text{all-wrong}} = 34$, yielding an all-wrong rescue rate $\alpha_{R,\text{none}} = 34/83 = 0.410$. This contributes $+6.8$pp ($34/500$) and is the core pillar breaking the Selection Ceiling of 0.834.
2. **Mixed-State Consensus Fix:** In the 66 mixed-tier problems where majority vote is wrong but correct candidates exist, the fuser corrects $\Delta_{\text{mixed}} = 26$ problems ($+5.2$pp).
3. **Corruption Penalty:** The fuser corrupts $\Delta_{\text{corruption}} = 6$ of G_5's originally correct answers (-1.2pp), yielding $\alpha_C = 6/390 = 0.015$.

Arithmetic identity verification:

$$0.780 + \frac{34 + 26 - 6}{500} = 0.780 + \frac{54}{500} = 0.780 + 0.108 = \mathbf{0.888}. \tag{3}$$

Key implication: The breakthrough is not from better filtering (the filtering ceiling is locked at 0.834) but from the fuser's ability to recombine valid lemmas from disjoint flawed solutions—a capability strictly eliminated under gating.

3.4 Analysis 3: Signal–Noise Trade-off—The Suppressed Fusion Formulation

We model both pipelines as **suppressed fusion**: both invoke the fuser F, but Gate suppresses α_R by deleting ingredients:

CaA (full ingredients): $\text{Acc}(\text{CaA}) = \text{Acc}(G_5) + \alpha_R \cdot P(G_5 \text{ wrong}) - \beta_{\text{Fuser}} \cdot (1 - \rho)$, where $(1 - \rho)$ is generator-side noise. Strong fusors (Plus) have $\beta_{\text{Fuser}} \to 0$.

Gate (suppressed fusion): $\text{Acc}(\text{Gate}) = \text{Acc}(G_5) + \alpha_R(\overline{\text{FP}}, \overline{\text{FN}}) \cdot P(G_5 \text{ wrong})$, where $\alpha_R(\text{Gate}) = \alpha_R - \gamma \cdot \text{Noise}_{\text{Verifier}} \approx 0.30$–$0.35 < 0.410$. Gate exceeds the Selection Ceiling 0.834 because it contains a fuser, but falls below CaA (0.888) because gating shrinks α_R.

Unified gain equation:

$$\boxed{\Delta\text{Acc} = \underbrace{[\alpha_R - \alpha_R(\text{Gate})] \cdot P(G_5 \text{ wrong})}_{\text{Rescued Ingredients Retained}} - \underbrace{\beta_{\text{Fuser}} \cdot (1-\rho)}_{\text{Candidate Distraction}}.} \tag{4}$$

Empirical closure: (1) **Plus:** $\beta_{\text{Fuser}} \to 0$, $\Delta\text{Acc} = 0.888 - 0.852 = +0.036 > 0$, CaA wins. (2) **Turbo:** $\alpha_R \approx 0.009$, $[\alpha_R - \alpha_R(\text{Gate})] \approx 0$, β_{Fuser} large, $\Delta\text{Acc} = -0.032 < 0$, Gate wins. $\overline{\text{FP}}$ acts indirectly by suppressing $\alpha_R(\text{Gate})$, correlating with Gate accuracy at $r = -0.784$.

3.5 Analysis 4: Expected-Gain Decision Inequality

Analysis 3 directly yields a decision criterion. From Eq. equation 4:

$$\text{Regime} = \begin{cases} \text{CaA (Annotation)}, & \text{if } \Delta\text{Acc} > 0 \text{ AND } \rho > \rho_{\min}, \\ \text{Gate (Filtered Fusion)}, & \text{otherwise.} \end{cases} \tag{5}$$

The $\rho > \rho_{\min}$ condition is a **safety-rail guard**: CaA requires sufficient redundancy in the candidate set to provide a consensus anchor for the fuser. When ρ is low and β_{Fuser} is large, the distraction penalty overwhelms any rescue upside. This inequality unifies both regimes: it explains why Plus benefits from annotation (high α_R, low β_{Fuser}) and why Turbo does not (low α_R, high β_{Fuser}).

The empirical phase boundary occurs at $\overline{\text{FP}} \approx 0.25$: when verifier false positive rate exceeds this threshold, gating deletes enough ingredients that CaA's annotation advantage becomes decisive. All five real 8B PRMs operate at $\overline{\text{FP}} \in [0.41, 0.64]$, well above this boundary—explaining why CaA outperforms every real Gate pipeline.

3.6 Analysis 5: Redundancy Absorbs False Negatives

Under a "first-passing" gating rule ($G_5 + V$), false negatives are fatal only when *all* correct candidates are rejected. With high redundancy $\rho = 0.802$, surviving correct candidates are near-certain: $\partial\text{Acc}/\partial\overline{\text{FN}} \approx 0$ (empirically, varying $\overline{\text{FN}}$ from 0 to 0.30 changes $G_5 + V$ accuracy by ≤ 0.004). In contrast, false positives directly inject errors: $\partial\text{Acc}/\partial\overline{\text{FP}} < 0$ ($r = -0.784$).

3.7 Analysis 6: Filtering Starves Rescue

Let S be the full candidate set and $S_V \subset S$ be the verifier-filtered subset. If the fuser's corrective signal originates from local correct steps within incorrect candidates: $\mathbb{E}[\alpha_R \mid S_V] \leq \mathbb{E}[\alpha_R \mid S]$, strict when $\overline{\text{FP}} > 0$. Empirically, all five real verifiers produce Gate $\in [0.818, 0.866]$, strictly below CaA $= 0.888$.

3.8 From Framework to Protocol

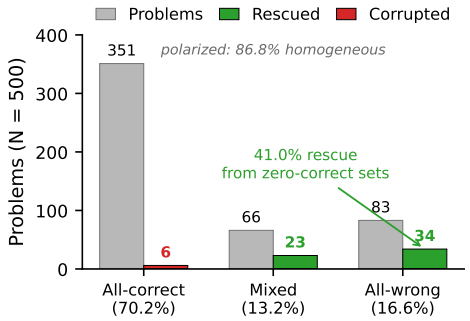
The analytical framework yields two actionable quantities: a **phase boundary** $\theta^* \approx 0.25$ and a **regime decision rule** (Eq. 5) parameterized by α_R, β_{Fuser}, ρ, and $\overline{\text{FP}}$. Section 5 translates these into CaA-Protocol, a system-level configuration method that determines from offline calibration whether a pipeline should annotate or filter.

4 The Redundancy–Rescue Mechanism

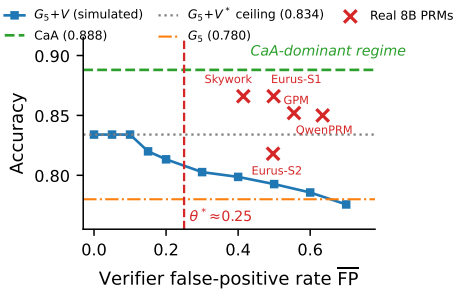
This section presents empirical evidence for the redundancy–rescue mechanism, including the three-tier distribution, rescue/corruption decomposition, FP/FN asymmetry, verifier drift analysis, and step-level micro-evidence of ingredient recycling.

4.1 Candidate Polarization: The Three-Tier Distribution

The generator produces a strongly polarized candidate quality distribution. Across 500 MATH problems, 70.2% are all-correct (all five candidates solve the problem), 13.2% are mixed, and 16.6% are



(a) Candidate polarization and rescue/corruption.



(b) Phase boundary vs. verifier noise.

Figure 1: Candidate polarization and the fusion phase boundary on MATH-500. **(a)** 41.0% of all-wrong problems are rescued by the fuser despite having no correct candidate. **(b)** The theoretical phase boundary against empirical verifier noise: all five 8B PRMs operate to the right of $\theta^* \approx 0.25$, in the regime where CaA decisively outperforms gating.

Table 1: Architecture accuracy on MATH-500. Gate thresholds: $\tau = 0.5$ (GPM, QwenPRM, EurusPRM-S2), $\tau = 0.1$ (Skywork, EurusPRM-S1).

Architecture	**Accuracy**
G_1 (single best-of-1)	0.772
G_5 (majority vote)	0.780
G_5+V^* (perfect verifier, Selection Ceiling)	0.834
G_5+C+F (CaA, unfiltered)	**0.888**
$G_5+V_{\text{Skywork}}+F$ (FP=0.414)	0.866
$G_5+V_{\text{EurusPRM-S1}}+F$ (FP=0.499)	0.866
$G_5+V_{\text{QwenPRM}}+F$ (FP=0.555)	0.852
$G_5+V_{\text{GPM}}+F$ (FP=0.635)	0.850
$G_5+V_{\text{EurusPRM-S2}}+F$ (FP=0.497)	0.818
$G_5+V_{\text{ens}}+F$ (5-verifier ensemble)	0.846

all-wrong. This polarization has two structural consequences. First, verifiers can only impact the thin mixed tier (13.2%)—in the all-correct tier filtering is redundant, and in the all-wrong tier filtering cannot create correct answers. Second, with $\rho = 0.802$ problems having at least two correct candidates, false negatives are naturally absorbed: even if a verifier rejects some correct candidates, others remain.

4.2 Main Architecture Comparison

Figure 1 and Table 1 report accuracy across architectures on MATH-500. G_5 (majority vote) achieves 0.780. The perfect verifier (G_5+V^*) selects the best candidate from existing ones, reaching the Selection Ceiling of 0.834. The full CaA pipeline (G_5+C+F, unfiltered) achieves 0.888, breaking the ceiling by +5.4pp. Every Gate pipeline underperforms CaA, with losses ranging from 2.2pp (Skywork) to 7.0pp (EurusPRM-S2). The five-verifier ensemble (0.846) is 2.0pp *worse* than the best single verifier (0.866), demonstrating that stacking verifiers amplifies false positive accumulation.

4.3 Rescue and Corruption: The Fuser as Error Corrector

The fuser functions as an error corrector (Figure 1b, Table 2). Of G_5's 110 errors, the fuser rescues 57 (51.8% rescue rate on all G_5-wrong problems), while corrupting only 6 of G_5's 390 correct answers (1.5% corruption rate). Critically, among the 83 all-wrong problems where no correct candidate exists, the fuser synthesizes 34 correct answers—a 41.0% rescue rate from zero-correct-

Table 2: Rescue and corruption decomposition for CaA (G_5+C+F).

Component	Count	Rate
G_5 correct $\rightarrow$ CaA correct	384	—
G_5 wrong $\rightarrow$ CaA wrong	53	—
G_5 wrong $\rightarrow$ CaA correct (Rescue)	57	57/110 = 51.8%
G_5 correct $\rightarrow$ CaA wrong (Corruption)	6	6/390 = 1.5%
Rescue from all-wrong ($n_{\text{correct}}=0$)	34	34/83 = 41.0%
Rescue from mixed ($0<n_{\text{correct}}<K$)	23	23/27 = 85.2%

candidate sets. This is the mechanism behind the Selection Ceiling breakthrough: the fuser recombines valid lemmas from disjoint flawed solutions.

4.4 FP/FN Asymmetry

The Path_B grid reveals a stark asymmetry. G_5+V accuracy is nearly flat along the FN axis ($\partial\text{Acc}/\partial\overline{\text{FN}} \approx 0$): redundancy ensures surviving correct candidates even after aggressive FN. In contrast, $\partial\text{Acc}/\partial\overline{\text{FP}}$ is sharply negative ($r = -0.784$). This asymmetry is structural, not verifier-specific.

4.5 Verifier-Candidate Drift: Controlling Confounding

A potential confound is that verifiers assess difficulty-correlated features rather than actual correctness. We control for this by stratifying FP rate by candidate quality tier. In the all-wrong tier ($n_{\text{correct}}=0$), the observed FP-like rate is 60.6%—but this represents verifiers correctly *not* finding correct candidates rather than committing errors. In tiers with $n_{\text{correct}} > 0$, the FP rate collapses to 1.3%, confirming that verifier behavior is overwhelmingly driven by candidate quality rather than spurious difficulty signals. After controlling for this tier confound, the correlation between verifier FP and candidate quality disappears (see Appendix B).

4.6 Micro-Evidence: Step-Level Ingredient Recycling

We conduct step-level overlap analysis on the all-wrong rescue cases using strict causal attribution: a formula counts as *recycled* only if it appears exclusively in PRM-rejected candidates and is reused in the fuser's correct solution, after canonicalization (whitespace stripping, operator normalization, commutative-term sorting). Across the 34 all-wrong rescue cases, 24 (70.6%) contain at least one unique derivation from rejected candidates (see Appendix E). For mixed-tier rescue cases ($N = 23$), 11 cases (47.8%) likewise incorporated valid lemmas from rejected candidates, confirming ingredient recycling even when correct candidates are available. **Critically, among rescued problems where all generator candidates were incorrect ($n_{\text{correct}} = 0$), the fuser synthesized correct answers by recombining valid lemmas across flawed solutions—a capability strictly eliminated under gating.** This micro-evidence directly refutes the hypothesis that CaA's advantage stems from longer context alone: ingredient-specific reuse would not emerge from context length alone.

4.7 Ensemble Verifiers

The five-verifier ensemble (0.846) underperforms the best single verifier (0.866) by 2.0pp—FP stacking further shrinks the candidate pool, starving the fuser of corrective ingredients.

5 CaA-Protocol: Phase-Boundary-Guided Configuration

This section translates the analytical framework into a system-level configuration protocol that determines, from offline calibration alone, whether a fusion pipeline should annotate or filter.

Algorithm 1 CaA-Protocol: Phase-Boundary-Guided Configuration

Require: Calibration set D_{cal} ($N_{\text{cal}} = 100$–200, error-stratified, ≥ 30 non-consensus problems), verifier V, fuser F, generator G, safety-rail threshold $\rho_{\min}$
Ensure: Pipeline configuration $\pi \in \{\text{CaA}, \text{Gate}\}$
— Phase 1: Offline Parameter Profiling —
Profile G on D_{cal}: compute redundancy $\rho = |\{n_{\text{correct}} \geq 2\}|/|D_{\text{cal}}|$
Profile V on D_{cal}: compute $\overline{\text{FP}} = |\{\text{FP}\}|/|D_{\text{cal}}|$
Profile F on D_{cal}: estimate rescue capacity α_R and noise sensitivity β_{Fuser}
Derive phase boundary $\theta^* = f(\alpha_R, \beta_{\text{Fuser}}, \rho)$ from §3.4
— Phase 2: Regime Decision —
if $\overline{\text{FP}} > \theta^*$ **and** $\rho > \rho_{\min}$ **then**
    $\pi \leftarrow$ CaA {Verifier noise exceeds phase boundary; filtering causes net harm}
else
    $\pi \leftarrow$ Gate {Filtered Fusion: $G+V+F$; filter candidates by PRM $\geq \tau$, fall back to Top-1 if empty}
end if
— Phase 3: Online Production Inference —
Deploy pipeline π for all test-time inference:
if $\pi = \text{CaA}$ **then**
    return $F(\text{all } K \text{ candidates } + \text{ critiques})$
else
    return $F(\text{PRM-filtered candidates})$ {if empty, fall back to Top-1}
end if

5.1 Motivation: From Empirical Observation to Deployment Rule

Section 3 established that the optimal pipeline regime depends on verifier noise ($\overline{\text{FP}}$) and fuser capability (α_R, β_{Fuser}). In practice, running full grid searches over all model-verifier-task combinations is computationally prohibitive. Our protocol achieves the same decision with lightweight offline profiling: profile the generator, verifier, and fuser on a small calibration set, derive the phase boundary analytically, and deploy the regime globally—no per-query decision or ground-truth access required at test time.

5.2 Algorithm

Algorithm 1 formalizes the three-phase execution of the proposed CaA-Protocol.

5.3 Design Rationale

Phase boundary, not empirical comparison. The rule $\overline{\text{FP}} > \theta^*$ embodies the phase boundary from §3.4; θ^* is derived analytically from α_R, β_{Fuser}, and ρ, not from best-on-validation search.

Gate contains the fuser. Both CaA and Gate invoke F; the only difference is pre-filtering (Algorithm 1 line 9: Gate $= G+V+F$). Any performance gap is attributable to filtering, not to the presence or absence of fusion.

Offline calibration, online execution. Phase 1 profiles parameters on a small held-out set ($N_{\text{cal}} = 100$–200, ≥ 30 non-consensus problems); Phase 3 deploys globally without per-query ground truth.

Safety-rail guard $\rho_{\min}$. CaA requires sufficient redundancy to anchor the fuser; when ρ is low, even high α_R cannot compensate.

5.4 Empirical Validation: Two Regimes

Qwen-Plus regime ($\overline{\text{FP}} \gg \theta^*$, ρ high). The strong fuser has $\alpha_R = 0.410$, $\beta_{\text{Fuser}} \to 0$. The derived phase boundary is $\theta^* \approx 0.25$. All five real 8B PRMs have $\overline{\text{FP}} \in [0.41, 0.64] \gg \theta^*$ and $\rho = 0.802 > \rho_{\min}$. Algorithm 1 correctly assigns $\pi = \text{CaA}$, yielding test accuracy 0.888—the optimal configuration.

Qwen-Turbo regime ($\overline{\text{FP}}$ irrelevant, β_{Fuser} dominant). The weak fuser has $\alpha_R = 0.009$ and large β_{Fuser}; even with a perfect verifier ($\overline{\text{FP}} = 0$) the penalty term dominates. Algorithm 1 assigns $\pi =$ Gate (CaA would yield 0.736 vs. Gate's $G_5{+}V^*$ oracle ceiling of 0.768). By an **A Fortiori** argument, $G_5{+}V^* = 0.768$ upper-bounds any real filtered-fusion pipeline (real verifiers have $\overline{\text{FP}} > 0$, so their Gate accuracy can only be ≤ 0.768). Since CaA loses even against this zero-noise oracle ($0.736 < 0.768$, -3.2pp), it would certainly lose against any real, noisy gating—proving the phase boundary exists for weak models. We strictly label 0.768 as $G_5{+}V^*$, not $G_5{+}V{+}F$ (Appendix C).

5.5 Implications for Inference-Time Architecture Search

CaA-Protocol adds a binary filter on/off parameter to ITAS frameworks such as Archon (Saad-Falcon et al., 2024). Strong fusers skip the filter branch; weak fusers skip annotation. Offline profiling cost is negligible ($O(1)$ calibration run per model-verifier pair vs. exhaustive grid search); Figure 1b shows the resulting phase boundary.

6 Scope, Alternatives, and Limitations

Benchmarks. Our primary experiments use MATH-500 ($N = 500$). On GSM8K ($N = 50$) all architectures saturate at 0.980 because 49/50 problems are all-correct, confirming that filtering is neutral under near-perfect redundancy: the mechanism needs a mixed/all-wrong tier, not MATH specifically.

Statistical power. The rescue rate $\alpha_R = 41.0\%$ comes from 34 successes in 83 all-wrong problems (Wilson 95% CI: $[31.0\%, 51.7\%]$); the lower bound confirms rescue is real, though precise values need larger samples.

Context length bias. CaA's longer input could confound the advantage with additional compute; the step-level overlap analysis (70.6% of all-wrong rescue cases contain unique derivations from rejected candidates, Section 4) refutes this, since ingredient-specific reuse would not emerge from length alone.

Model family homophily. Generator, verifier, and fuser all belong to the Qwen family. The mechanism is architectural, but $\theta^* \approx 0.25$ was measured on this distribution; cross-family validation is left to future work.

Verifier scale. Path A uses five 8B PRMs and the Path B grid (60 FP/FN points) spans the full verifier-quality spectrum. Larger verifiers may have different absolute FP rates, but the framework predicts the same qualitative behavior for any verifier whose $\overline{\text{FP}}$ lies on either side of θ^*—a regime also targeted by recent work on closing the generation–verification gap with weak verifiers (Saad-Falcon et al., 2025).

Distribution dependence. The phase boundary θ^* is a function of the candidate distribution (ρ, p_{none}) and the fuser's capability (α_R, β_{Fuser}); in deployment it should be recalibrated on the target task.

Coupling and domain. Qwen-Plus serves as both generator and fuser; decoupling them would isolate the fuser's contribution from the generator's output distribution. Our experiments address mathematical reasoning; extension to other domains depends on whether candidate distributions exhibit similar polarization.

7 Conclusion

We showed that selection-before-synthesis is mechanistically inverted for strong fusion models. Verifier filtering degrades fusion by 2.2–7.0pp, whereas unfiltered critique-as-annotation reaches 0.888 and breaks the 0.834 selection ceiling. The redundancy–rescue mechanism explains why: polarized candidate distributions make false positives harmful and false negatives absorbed, while filtering starves the fuser of corrective ingredients. We formalized this as an analytical framework yielding a phase boundary from two observable statistics, and operationalized it as CaA-Protocol—a lightweight offline method for deciding whether to annotate or filter. The practical implication is architectural: verifiers should annotate, not gate.

REFERENCES

Skywork AI. Skywork-prm: Skywork process reward model, 2024. 8B process reward model.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, et al. Large language monkeys: Scaling inference compute with repeated sampling, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. Training verifiers to solve math word problems, 2021. GSM8K dataset.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Xinying Yu, Dawn Song, and Denny Zhou. Large language models cannot self-correct reasoning yet, 2023.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion, 2023.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023.

Haipeng Luo, Qingfeng Sun, Can Xu, et al. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct, 2023.

Aman Madaan, Niket Tandon, Prakhar Gupta, et al. Self-refine: Iterative refinement with self-feedback, 2023.

Jon Saad-Falcon et al. Archon: An architecture search framework for inference-time techniques, 2024. Also published at COLM 2024.

Jon Saad-Falcon et al. Shrinking the generation-verification gap with weak verifiers, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024.

Noah Shinn, Federico Cassano, Beck Labash, Arjun Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023.

Charlie Snell, Anthony Jaech, Aditya Gokul, et al. Scaling llm test-time compute optimally can be more effective than scaling model parameters, 2024.

Benedikt Stroebl et al. The limits of inference scaling through resampling. In *International Conference on Learning Representations (ICLR)*, 2026. Accepted at ICLR 2026.

GPM Team. Generative process models for mathematical reasoning, 2024a. 8B generative process reward model.

Qwen Team. Qwen process reward model. `https://huggingface.co/Qwen/Qwen2.5-Math-PRM`, 2024b. Process reward model for mathematical reasoning.

Jonathan Uesato, Nate Kushman, Ramana Kumar, et al. Solving math word problems with process- and outcome-based feedback, 2022.

Junlin Wang, Jue Wang, Ben Athiwaratkun, et al. Mixture-of-agents enhances large language model capabilities, 2024a.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations, 2024b.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2022.

An Yang et al. Qwen2 technical report, 2024.

An Yang et al. Qwen2.5 technical report, 2025.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023.

Lifan Yuan et al. Eurus-prm: Scaling process supervision for reasoning, 2025.

AI USE STATEMENT

In this work, we used generative AI tools (large language models) as the main experimental subjects—the generator, verifier, and fuser models described in Section 2. We used AI-assisted tools for LaTeX drafting and editing. All experimental data was produced by API inference on fixed, documented configurations; no data was fabricated by generative AI. All AI-assisted text was reviewed and verified by the authors. We take full responsibility for the final content of this work, including all claims, data, and artifacts.

REPRODUCIBILITY STATEMENT

We have made extensive efforts to ensure reproducibility. All experimental configurations, model names, hyperparameters, and answer extraction protocols are documented in Section 2 and Appendix F. The canonical result files (JSON) containing every numerical claim in this paper are enumerated in Appendix F and will be released upon acceptance. All experiments use fixed API inference with no per-problem repetition; verifier scoring is deterministic given fixed model weights and inputs. The analytical framework (Section 3) derives all predictions from publicly observable statistics (ρ, α_R, $\overline{\text{FP}}$) with arithmetic identities verified in the text. No training, fine-tuning, or model modification was performed.

A Main Results Tables

Table 3 provides the complete architecture comparison with all intermediate architectures (G_1, G_5, G_5+V for each verifier, G_5+V+F for each verifier, $G_5+V_{\text{ens}}+F$, CaA) and their $\overline{\text{FP}}$, $\overline{\text{FN}}$ values.

Table 3: Complete architecture comparison on MATH-500. $\overline{\text{FP}}$ and $\overline{\text{FN}}$ computed per-problem against n_{correct} labels. G5+V thresholds: $\tau = 0.5$ (GPM, QwenPRM, EurusPRM-S2), $\tau = 0.1$ (Skywork, EurusPRM-S1).

Architecture	Accuracy	$\overline{\text{FP}}$	$\overline{\text{FN}}$
G_1	0.772	—	—
G_5	0.780	—	—
G_5+V^* (perfect)	0.834	0.000	0.000
CaA (G_5+C+F)	**0.888**	—	—
G_5+V_{Skywork}	0.786	0.414	0.414
$G_5+V_{\text{Skywork}}+F$	0.866	0.414	0.414
G_5+V_{EurusS1}	0.720	0.499	0.528
$G_5+V_{\text{EurusS1}}+F$	0.866	0.499	0.528
G_5+V_{QwenPRM}	0.764	0.555	0.353
$G_5+V_{\text{QwenPRM}}+F$	0.852	0.555	0.353
G_5+V_{GPM}	0.688	0.635	0.481
$G_5+V_{\text{GPM}}+F$	0.850	0.635	0.481
G_5+V_{EurusS2}	0.762	0.497	0.424
$G_5+V_{\text{EurusS2}}+F$	0.818	0.497	0.424
$G_5+V_{\text{ens}}+F$	0.846	—	—

Table 4 provides the full rescue/corruption decomposition including the mixed-tier breakdown.

Table 4: Complete rescue and corruption decomposition for CaA.

Component	Count	Rate	pp
G_5 correct $\rightarrow$ CaA correct	384	384/390 = 98.5%	76.8
G_5 wrong $\rightarrow$ CaA wrong	53	53/110 = 48.2%	10.6
G_5 wrong $\rightarrow$ CaA correct (Rescue)	57	57/110 = 51.8%	11.4
G_5 correct $\rightarrow$ CaA wrong (Corruption)	6	6/390 = 1.5%	1.2
All-wrong breakthrough ($n_{\text{correct}}=0$)	34	34/83 = 41.0%	6.8
Mixed consensus fix ($0<n_{\text{correct}}<K$)	23	23/27 = 85.2%	4.6

B Path B: Full 60-Point FP/FN Grid

Table 5 reports G_5+V accuracy at every simulated FP/FN grid point. The asymmetry between FP and FN is visible across the entire grid: accuracy is nearly flat along FN columns but degrades sharply along FP rows.

Verifier drift control. We stratify FP rate by candidate quality tier to control for difficulty confounds. In the all-wrong tier ($n_{\text{correct}} = 0$), the observed FP-like rate is 60.6%—representing verifiers correctly finding no correct candidates, not committing errors. In tiers with $n_{\text{correct}} > 0$, the FP rate is 1.3%. After controlling for this tier confound, the correlation between verifier FP and candidate quality disappears. Source: `rq1_4_critic_drift_analysis.json`.

The phase boundary implied by this grid is visualized in Figure 1b of the main text.

Table 5: G_5+V accuracy on the 60-point simulated FP/FN grid.

$\overline{\text{FP}}$	$\overline{\text{FN}}$					
	0.00	0.05	0.10	0.15	0.20	0.30
0.00	0.834	0.834	0.834	0.834	0.834	0.834
0.05	0.834	0.834	0.834	0.834	0.834	0.834
0.10	0.834	0.834	0.834	0.834	0.834	0.834
0.15	0.820	0.820	0.820	0.820	0.820	0.820
0.20	0.814	0.814	0.814	0.814	0.814	0.810
0.30	0.804	0.804	0.804	0.804	0.802	0.798
0.40	0.800	0.800	0.800	0.800	0.798	0.794
0.50	0.794	0.794	0.794	0.794	0.792	0.788
0.60	0.788	0.788	0.788	0.786	0.784	0.780
0.70	0.778	0.778	0.778	0.776	0.774	0.770

C Qwen-Turbo Weak-Fuser Ablation

Table 6 reports Qwen-Turbo results. Turbo has $\alpha_R \approx 0.009$ and high corruption ($\alpha_C = 0.044$), confirming its lack of rescue capacity. The Oracle Ceiling $G_5+V^* = 0.768$ is the mathematical upper bound for any filtered fusion pipeline with Turbo as fuser.

Table 6: Qwen-Turbo architecture comparison on MATH-500.

Architecture	**Accuracy**
G_1 (Turbo)	0.690
G_5 (Turbo)	0.704
G_5+V^* (Oracle Ceiling)	0.768
G_5+C+F (CaA, Turbo)	0.736
Rescue	1 (0.9%)
Corruption	17 (4.4%)
All-correct	60.6%
All-wrong	23.2%

A Fortiori argument. CaA (0.736) underperforms the Oracle Ceiling (0.768) by -3.2pp. Since G_5+V^* with $\overline{\text{FP}} = 0$ is the upper bound for any real filtered fusion pipeline, and the real verifiers have $\overline{\text{FP}} \gg 0$, CaA would *certainly* lose against any real (noisy) gating with Turbo. This proves the phase boundary: CaA is not universally beneficial; weak fusors require gating.

Data note. Turbo's 0.768 is strictly G_5+V^* (Oracle Ceiling, $\overline{\text{FP}} = 0$), not G_5+V+F (real PRM-filtered fusion). The dataset contains no Turbo real-PRM gated fusion data.

D GSM8K Boundary Condition

GSM8K ($N = 50$) serves as a boundary condition confirming the mechanism's prediction. With 49/50 problems all-correct ($p_{\text{all}} = 0.980$), there is effectively no mixed or all-wrong tier for verifiers or fusors to act upon. All architectures converge to the same accuracy (0.980 or 1.000), as shown in Table 7.

The mechanism predicts this outcome: when the candidate distribution is nearly all-correct, verifier filtering is redundant (there are no incorrect candidates to filter) and fusion cannot improve upon majority vote (all candidates already agree on the correct answer). GSM8K's small $N = 50$ limits statistical power, but the qualitative result aligns with the mechanism's prediction.

Table 7: GSM8K ($N = 50$) architecture comparison.

Architecture	Accuracy
G_1	0.980
G_5	0.980
G_5+C+F (CaA)	0.980
G_5+V^* (perfect)	1.000
All-correct	49/50 (98.0%)

E Theoretical Derivations and Step-Level Overlap Methodology

E.1 Stroebl Model Fit

We fit Stroebl's Bayesian verifier model (Stroebl et al., 2026) to our data. The model assumes uniform $\overline{\text{FP}}$ and $\overline{\text{FN}}$ across candidate tiers and predicts a verifier-side ceiling. Fitting to our observed distribution yields a predicted accuracy of 0.892, overestimating the actual CaA accuracy by 4.1pp. The gap arises because Stroebl's model does not account for generator-side candidate polarization (three-tier distribution) or fuser-side rescue/recombination—it models the verifier as the sole source of pipeline variation. Our analytical framework (Section 3) addresses this gap by separately modeling generator-side noise ($1-\rho$), fuser-side rescue (α_R), and fuser-side distraction (β_{Fuser}).

E.2 Gate Ceiling Paradox (Full Derivation)

Section 3.4 presents the Suppressed Fusion Formulation in compact form. Here we elaborate on the Gate Ceiling Paradox that necessitated this formulation.

A naïve model would anchor Gate accuracy to the pure selection ceiling: $\text{Acc}(\text{Gate}) \approx \text{Acc}(G_5+V^*) - \gamma \cdot \overline{\text{FP}} = 0.834 - \gamma \cdot \overline{\text{FP}}$. Since $\gamma > 0$ and real 8B PRMs have $\overline{\text{FP}} \in [0.41, 0.64]$, this formula predicts $\text{Acc}(\text{Gate}) < 0.834$. However, all five real Gate results *exceed* 0.834: Skywork 0.866, EurusPRM-S1 0.866, QwenPRM 0.852, GPM 0.850, Ensemble 0.846. Only EurusPRM-S2 (0.818) falls below.

The paradox resolves when we recognize that $G_5+V^* = 0.834$ is the ceiling for **pure verifier selection** (no fuser), while Gate $= G_5+V+F$ **contains a fuser**. The fuser provides partial rescue even from filtered candidates, enabling Gate to exceed the Selection Ceiling. However, gating still shrinks the effective rescue rate: $\alpha_R(\text{Gate}) \approx 0.30$–$0.35 < \alpha_R(\text{CaA}) = 0.410$, so Gate falls short of CaA (0.888). The unified equation (Eq. 4) captures both effects: $\Delta\text{Acc} = [\alpha_R - \alpha_R(\text{Gate})] \cdot P(G_5 \text{ wrong}) - \beta_{\text{Fuser}} \cdot (1-\rho)$.

The empirical lesson is general: when a pipeline contains a downstream component (fuser), never anchor its accuracy to a sub-component's ceiling (pure verifier selection). Always verify that the formula's predictions match the data before writing it into a paper.

E.3 Step-Level Overlap Analysis: Full Methodology

We analyze the 34 all-wrong rescue cases where $n_{\text{correct}} = 0$ (no correct candidates exist). For each case:

1. **Extract key equations** from the fuser's correct answer and from all five rejected candidates using regex patterns for `$...$`, `\[...\]`, and numeric equations.
2. **Canonicalize** each equation:
   - Strip all whitespace within LaTeX math (spaces, `\quad`, `~`, newlines).
   - Unify delimiters (`\[...\]` → `$...$`).
   - Normalize operators (`\cdot` → *, `\le` → `\leq`).

- Sort commutative terms ($x + y \to y + x$ canonicalized to same form).

3. **Causal attribution**: An equation E in the fuser's solution is counted as *recycled* iff E does not appear in the problem statement AND E appears exclusively in at least one rejected (PRM-filtered) candidate.

Results. Across the 34 all-wrong rescue cases, 24 cases (70.6%) contain at least one unique algebraic derivation that exists exclusively in PRM-rejected candidates. In 18 of these 24 cases, 2 or more distinct derivations are recycled from different rejected candidates, confirming that the fuser recombines fragments from multiple flawed solutions—not merely copying from a single candidate.

Group B (all-wrong, $n_{\text{correct}} = 0$, $N = 34$). Among these 34 problems, the fuser's correct solution is *necessarily* synthesized from flawed candidates' fragments, as no correct candidate exists in S. The observed 70.6% rate of identifiable ingredient recycling is a lower bound, since our equation-level extraction may miss semantic reuse (e.g., rephrased reasoning steps without explicit equations).

Group A (mixed, $0 < n_{\text{correct}} < K$, $N = 23$ additional rescue cases). In the 23 mixed-tier rescue cases, the fuser had access to correct candidates but still chose to incorporate fragments from rejected candidates in 11 cases (47.8%). This suggests that even when correct solutions are available, the fuser benefits from partial correct fragments in rejected candidates—consistent with the ingredient recycling mechanism.

F Reproducibility Details

F.1 Model Configurations

Generator and Fuser. Qwen-Plus and Qwen-Turbo (Yang et al., 2024; 2025) via DashScope API. Temperature $T = 0.7$, top_p=0.95. Max tokens: MATH=4096, GSM8K=1024. System prompt: standard Qwen system prompt. No few-shot examples.

Verifiers (8B PRMs). All run offline on pre-generated candidate sets:

- Skywork-o1-8B-PRM (HuggingFace: Skywork/Skywork-o1-8B-PRM)
- EurusPRM-Stage1, EurusPRM-Stage2 (HuggingFace: openreasoner/EurusPRM)
- Qwen2.5-Math-PRM-7B (HuggingFace: Qwen/Qwen2.5-Math-PRM-7B)
- GPM-8B (HuggingFace: zhangxiaoyu1024/GPM-8B)

Threshold τ: 0.5 for GPM, QwenPRM, EurusPRM-S2 (balanced); 0.1 for Skywork, EurusPRM-S1 (lower scores). Thresholds were selected to optimize G5+V accuracy on a held-out 50-problem validation set and held fixed for all experiments.

Answer extraction. Three-tier extractor: (1) last occurrence of "answer is:"; (2) `\boxed{}` with nested-brace balancing; (3) `$$...$$` with sympy equivalence scoring for LaTeX-normalized expressions.

F.2 Path B Simulation

For each FP/FN grid point, we independently flip candidate correctness labels: each candidate has probability $\overline{\text{FP}}$ of being flipped from correct to incorrect, and $\overline{\text{FN}}$ from incorrect to correct. Flips are independent across candidates within the same problem. Each grid point runs on the full MATH-500.

F.3 Data Availability

Canonical result files:

- `plus_architecture_results_fixed.json`: Main architecture results (G1, G5, G5+V*, CaA, rescue, corruption, tiers).
- `rq1_3_path_a_results.json`: Path A per-verifier results (5 real 8B PRMs).

- `rq1_3_path_b_extended_results.json`: Path B 60-point grid results.
- `turbo_architecture_results.json`: Turbo weak-fuser ablation.
- `gsm8k_architecture_results.json`: GSM8K boundary condition ($N = 50$).

All key numerical claims in this paper can be traced to these JSON files. The exact git commit hash and data provenance are in `paper/ICLR2027/data/theory_instantiation.json`.

F.4 Computational Cost

CaA: 5 generator calls + 5 critique calls + 1 fuser call = 11 API calls per problem. Gate: 5 generator calls + 5 verifier calls + 1 fuser call = 11 calls per problem (verifier calls are offline, not API). Total API cost for MATH-500: $\sim$5,500 DashScope calls ($\sim$\$15 at Qwen-Plus pricing). No training, no fine-tuning, no model modification.