SEVA: Self-Evolving Verification Agent with Process Reward for Fact Attribution

Anonymous Authors[1]

Abstract

Hallucination remains a critical reliability bottleneck for LLM-based agents, and fact attribution verifiers serve as the last line of defense. Yet current verifiers produce opaque binary labels, revealing nothing about *which* evidence was considered or *why* the judgment was made — leaving agents unable to correct themselves and operators unable to audit decisions. We introduce SEVA (**S**elf-**E**volving **V**erification **A**gent), which outputs structured evidence alignments, step-by-step reasoning chains, calibrated confidence, and error diagnoses from a six-category taxonomy. To train SEVA beyond SFT, we design a process reward function that scores five verification components independently, allocating 70% of weight to process signals (evidence grounding, reasoning quality, format compliance) and 30% to outcome (label, diagnosis). This design is not optional: we show that standard binary reward suffers from *advantage collapse* on structured output, where GRPO receives no gradient signal and training stagnates entirely. Process reward resolves this and induces an implicit curriculum — the agent first masters verification *behavior* (alignment: $0.917 \rightarrow 0.997$; format: $72\% \rightarrow 100\%$), then improves verification *outcomes* (F1: $64.9 \rightarrow 69.0$). Structured output further enables a self-evolution loop where the agent's own errors — now transparent and diagnosable — drive targeted adversarial data generation for iterative improvement. On ClearFacts, SEVA-3B matches GPT-4o-mini (69.0 vs. 69.8 F1) while producing substantially richer, auditable output.

1. Introduction

Despite rapid progress in LLM capabilities, hallucination remains a fundamental barrier to deploying agents in high-stakes domains such as finance, law, and healthcare (Min et al., 2023). Fact attribution verifiers — models that judge whether each claim in an agent's output is supported by its source documents — have emerged as a critical safety layer (Gao et al., 2023; Tian et al., 2024). Systems like MiniCheck (Tang et al., 2024) and ClearCheck (Seo et al., 2025) achieve strong accuracy on this task, but they share a fundamental limitation: they produce only a binary label.

This opacity creates two problems for agents in the wild. First, when a verifier flags a claim, the agent has no basis for *self-correction* — it knows something is wrong but not what. Was the percentage inflated? The entity swapped? A qualifier fabricated? Second, no human operator can meaningfully *audit* the decision, because the reasoning behind the label is invisible. In safety-critical deployments, an uninterpretable verifier undermines the very trust it is meant to provide.

We introduce SEVA (**S**elf-**E**volving **V**erification **A**gent), which addresses both problems by producing *structured* verification output: evidence alignment spans that ground every judgment in specific text, reasoning chains that trace the logic step by step, and error diagnoses from a six-category taxonomy with actionable fix suggestions. This structured output serves a dual purpose: it makes verification auditable for deployment, and it provides a diagnostic interface for training.

How should such an agent be trained? SFT on teacher-annotated data provides a reasonable starting point, but reinforcement learning — which has driven substantial gains for mathematical reasoning (Shao et al., 2024a; Zha et al., 2025) and hallucination reduction (Li et al., 2026) — does not straightforwardly transfer. Applying GRPO (Shao et al., 2024a) with binary reward (1 if the label matches, 0 otherwise) to our structured verifier, we find that training stalls entirely: the policy makes no progress beyond SFT across all 350 steps. The culprit is that binary reward compresses all verification quality into a single bit. A response with correct reasoning but the wrong label receives the same

[1]**AUTHORERR: Missing \icmlaffiliation. .AUTHORERR: Missing \icmlcorrespondingauthor.**

Preliminary work. Under review at the Second Workshop on Agents in the Wild: Safety, Security, and Beyond (AIWILD) at ICML 2026. Do not distribute.

score — zero — as one that produces unparseable garbage. In a GRPO group of G=8 responses, most therefore score 0, advantage spread contracts to ± 0.05, and the gradient vanishes.

Our response is a **process reward function** that decomposes verification quality into five independently scored components:

$$R = \underbrace{w_f R_f + w_a R_a + w_c R_c}_{\text{process (70\%)}} + \underbrace{w_l R_l + w_d R_d}_{\text{outcome (30\%)}} + R_{\text{cal}} \quad (1)$$

Process components — format R_f, alignment R_a, chain R_c — receive 70% of total weight, directing the reward signal toward verification *quality* rather than just the final label. Under this scheme, a response with sound reasoning but the wrong label scores $\sim$0.63 rather than 0.0, ensuring $\sigma > 0$ in Eq. 2 and restoring meaningful policy gradients.

The results confirm that process reward unlocks what binary reward cannot. GRPO lifts alignment quality to 0.997, format compliance to 100%, and F1 to 69.0 (+4.1 over SFT). An implicit curriculum emerges in the training dynamics: the agent masters verification *behavior* within 150 steps, then spends the remaining 200 steps refining verification *outcomes* — without any explicit scheduling.

Structured output confers one further advantage: it makes the agent's failures *transparent*. When SEVA misclassifies a claim, its evidence alignments and error diagnoses pinpoint what went wrong — which error category was missed, where grounding broke down. We channel this diagnostic signal into a **Verify→Reflect→Probe→Refine** self-evolution loop that generates targeted adversarial data for the agent's weakest error categories, enabling the kind of surgical improvement that binary verifiers, with their opaque outputs, simply cannot support.

Our contributions:

1. **SEVA**: a structured verification agent that produces auditable evidence alignments, reasoning chains, and error diagnoses, trained via process-reward GRPO to match GPT-4o-mini at 3B scale (§2.2, §3).
2. A **process reward function** that resolves the advantage collapse inherent in binary reward for structured output, and reveals an implicit curriculum effect (§2.3, §3.5).
3. A **self-evolution loop** that leverages structured output as a diagnostic interface for targeted adversarial data generation (§2.5).

2. Method

2.1. Problem Formulation and Overview

Given a claim c and source document d, a fact attribution verifier produces a judgment about whether c is supported by d. Existing verifiers output a single binary label (Tang et al., 2024; Seo et al., 2025). We instead require the agent to produce structured output $\mathbf{v} = (A, C, y, \gamma, e, s)$ that makes verification auditable and failures diagnosable (Figure 1).

Building such an agent via SFT is straightforward, but pushing it further with RL is not. We show that GRPO with binary reward fails entirely on this output format (§2.3), and design a process reward that resolves this failure (§2.3). The structured output further enables a self-evolution loop for iterative improvement (§2.5).

2.2. Structured Verification Schema

The output $\mathbf{v}$ comprises four complementary components:

Evidence alignment A: a list of $(c_i, d_i, \text{status}_i)$ triples mapping claim spans to source spans, with status $\in$ {match, mismatch, not_found}. Each entry forces the agent to anchor its judgment in specific text rather than forming a holistic impression.

Reasoning chain C: step-by-step verification where each step examines a claim part against source evidence, producing a judgment $\in$ {supported, not_supported, partially_supported} and a natural language explanation.

Label and confidence: a binary label y paired with calibrated confidence $\gamma \in [0, 1]$.

Error diagnosis: when y=Not Attributable, an error type e drawn from a six-category taxonomy (numerical exaggeration, negation flip, scope inflation, temporal shift, entity substitution, fabrication) together with a fix suggestion s.

This schema serves two purposes relevant to agents in the wild. First, it makes verification *auditable*: a human operator can inspect alignments and reasoning to judge whether the verdict is trustworthy. Second, it makes failures *diagnosable*: when the agent errs, the structured output pinpoints which evidence was mishandled, feeding the self-improvement loop in §2.5.

2.3. From Binary Reward Failure to Process Reward

The failure of binary reward. Binary reward assigns 1.0 when the predicted label matches the gold label and 0.0 otherwise. For structured output, this produces a degenerate training signal. In a GRPO group of G=8 responses: (1) 28% fail JSON parsing — the SFT model produces valid JSON only 72% of the time, and all of these score 0; (2) among valid responses, $\sim$35% predict the wrong label, also scoring 0; (3) in a typical group, 5–7 of 8 responses receive zero reward. GRPO computes advantages relative to the group mean. For a group of G responses with rewards $\{r_1, \ldots, r_G\}$, the normalized advantage of response i is:

$$\hat{A}_i = \frac{r_i - \mu}{\sigma + \epsilon}, \quad \mu = \frac{1}{G}\sum_j r_j, \quad \sigma = \text{std}(\{r_j\}) \quad (2)$$

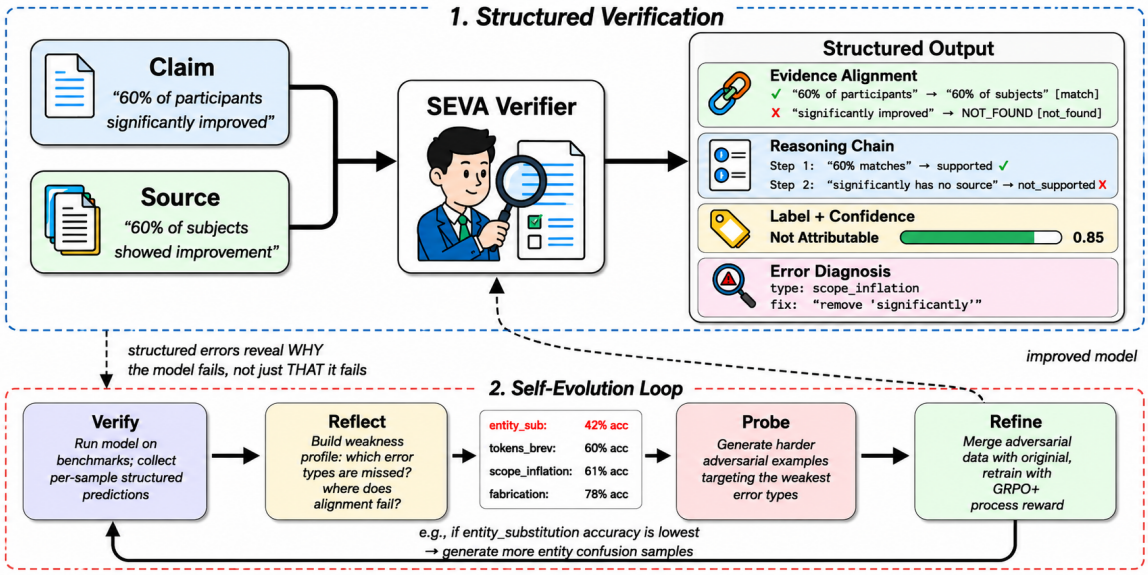

Figure 1. **SEVA overview.** *Top:* Given a claim-source pair, the verifier produces structured output — evidence alignments, reasoning chains, calibrated confidence, and error diagnosis. *Bottom:* Self-evolution loop. Structured errors reveal *why* the model fails (not just *that* it fails), enabling targeted adversarial data generation focused on the weakest error types.

When binary reward produces $r_j \in \{0, 1\}$ with most $r_j = 0$, both μ and σ are near zero, and $\hat{A}_i \approx 0$ for all i — the policy gradient $\nabla_\theta J \propto \sum_i \hat{A}_i \nabla_\theta \log \pi_\theta$ vanishes regardless of model parameters.

This failure is *structural*, not incidental. Increasing group size does not help — the problem is near-uniform scores, not insufficient sampling. And the failure is not specific to verification: any RL task whose output has multiple required components will exhibit advantage collapse under binary reward whenever the model cannot reliably produce all components simultaneously.

Process reward as the resolution. We resolve advantage collapse by replacing the single binary signal with five component-wise scores (Eq. 1). Weights allocate 70% to process components (w_f=0.10, w_a=w_c=0.30) and 30% to outcome components (w_l=w_d=0.15), plus an asymmetric calibration term R_{cal}.

The central design principle is that sound verification reasoning should be rewarded even when the final label is wrong. Table 1 makes this concrete: "good reasoning, wrong label" scores 0.63 under process reward but 0.0 under binary. Conversely, "correct label, poor reasoning" scores only 0.28 vs. 1.0 — binary reward effectively rewards lucky guesses over genuine understanding, while process reward inverts this ranking.

Why 70/30? The weight split reflects a deliberate design choice: in early training, the model must first learn to produce well-formed structured output before it can meaningfully improve its label predictions. If outcome components dominated (e.g., 70% label + 30% process), a model that guesses the correct label but produces incoherent reasoning would score highly — learning to "game" the reward rather than genuinely verify. By weighting process at 70%, we ensure that the only way to achieve high reward is through substantive verification work: grounding claims in source text, building coherent reasoning chains, and producing valid structured JSON. The label then serves as a refinement signal on top of already-good verification behavior.

Table 1. Reward landscape. Process reward produces four distinct quality levels where binary reward sees only two, enabling fine-grained advantage estimation within each GRPO group.

Response quality	Process	Binary
Correct label + good reasoning	∼1.13	1.0
Good reasoning, wrong label	∼0.63	0.0
Correct label, poor reasoning	∼0.28	1.0
Unparseable output	0.0	0.0

Component scoring. Each component is scored independently (full rubrics in Appendix C). R_f checks format compliance: valid JSON with all required fields. R_a evaluates each evidence alignment entry on span presence, source grounding, status validity, and span length. R_c evaluates each reasoning step on judgment validity, explanation quality, evidence citation, and claim coverage, with a bonus for multi-step chains. R_d rewards correct error-type identification and actionable fix suggestions for negative predictions. R_{cal} adds $+\gamma \times 0.15$ when the label is correct, encouraging

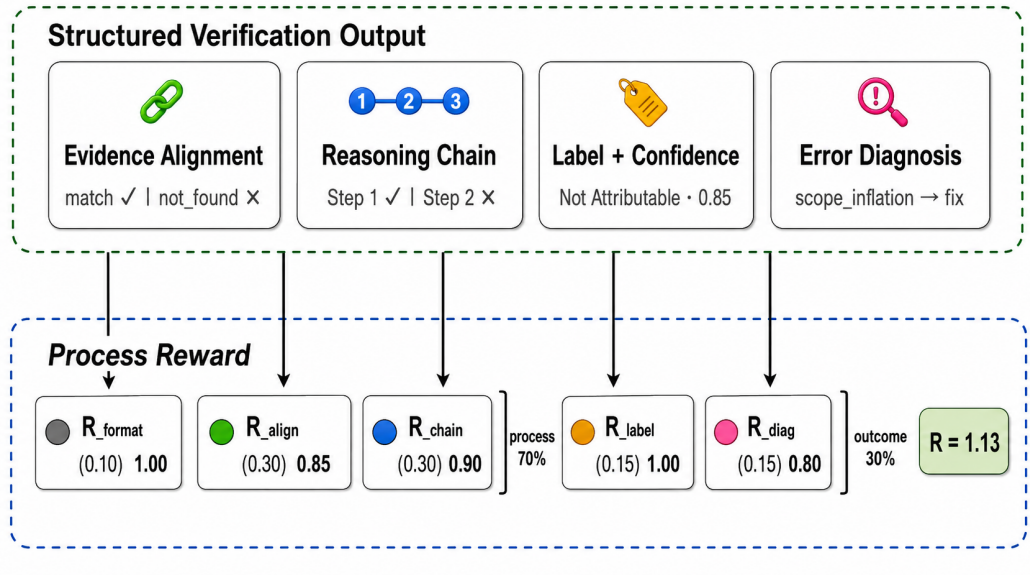

Figure 2. **Process reward scoring.** Each structured output component (left) maps to an independently scored reward term (right). A response with correct reasoning but the wrong label scores 0.63 under process reward vs. 0.0 under binary — this gap is what provides GRPO with meaningful gradients.

Algorithm 1 Process Reward Computation

Require: Response r, gold label y^*
Ensure: Reward $R \in [-0.10, 1.28]$
Parse r as JSON $\rightarrow \hat{v}$
if parse fails **then**
  $R = 0.0$
end if
$R_f \leftarrow \text{ScoreFormat}(\hat{v})$ {0 / 0.2 / 0.5 / 1.0}
$R_a \leftarrow \frac{1}{|A|} \sum_{a \in A} \text{ScoreAlign}(a)$
$R_c \leftarrow \frac{1}{|C|} \sum_{s \in C} \text{ScoreStep}(s) + \text{LenBonus}$
$R_l \leftarrow \mathbf{1}[\text{normalize}(\hat{y}) = y^*]$
$R_d \leftarrow \text{ScoreDiagnosis}(\hat{e}, \hat{s}, y^*)$
$R \leftarrow 0.10\, R_f + 0.30\, R_a + 0.30\, R_c + 0.15\, R_l + 0.15\, R_d$
if $\hat{y} = y^*$ **then**
  $R \leftarrow R + \hat{\gamma} \times 0.15$ {calibration bonus}
else
  $R \leftarrow R - \hat{\gamma} \times 0.10$ {overconfidence penalty}
end if$R$

calibrated confidence, and subtracts $\gamma \times 0.10$ when wrong, penalizing overconfidence. Algorithm 1 summarizes the full computation.

2.4. Training Pipeline

SEVA is trained in two phases. In the first, we generate 4,992 structured annotations for ANLI (Nie et al., 2020) examples using GPT-4o-mini as teacher (92% pass format validation) and fine-tune Qwen2.5-3B-Instruct (Qwen Team, 2025) on these annotations for 3 epochs (lr$= 2 \times 10^{-5}$, full parameters). This SFT checkpoint serves both as our baseline and as the starting policy for RL.

In the second phase, we apply GRPO with the process reward. Each prompt generates G=8 responses at temperature 1.2 (top-p=0.95). Training runs for 5 epochs ($\sim$350 steps) at lr$= 2 \times 10^{-6}$ with KL coefficient β=0.001, using veRL (Shao et al., 2024b) with FSDP on 2×RTX 6000 Ada GPUs. Total compute including SFT is $\sim$28 GPU-hours.

Full hyperparameters are in Appendix A.

Two design choices in the SFT-to-GRPO transition matter for reproducibility. We use a GRPO learning rate of 2×10^{-6} (10× lower than SFT) to preserve the structured output format that SFT established. The KL coefficient β=0.001 is also kept small: too much KL regularization would prevent the policy from exploring the reward landscape, while too little would allow reward hacking. We found this balance after observing that β=0.01 caused the model to stay too close to the SFT policy (F1 improvement $<$1 point), while β=0 led to occasional degenerate outputs where the model exploited the format reward by producing well-structured but semantically vacuous responses.

We additionally train a 7B variant via two-stage SFT: first on 59K binary NLI samples for broad classification ability, then on 5K structured samples for the output format. This model uses LoRA (Hu et al., 2022) (rank 128) due to memory constraints; 7B full fine-tuning with GRPO is ongoing.

2.5. Self-Evolution via Structured Diagnostics

Structured output provides a diagnostic interface unavailable to binary verifiers: when the agent misclassifies a claim, evidence alignments and error diagnoses reveal *which* aspect

of verification failed. Inspired by the multi-agent decomposition in MARCH (Li et al., 2026), where a Solver generates claims, a Proposer decomposes them, and a Checker validates against evidence in isolation, we adopt a similar principle of *functional separation* — not across agents, but across stages of the improvement process.

Our **Verify→Reflect→Probe→Refine** loop (Figure 1, bottom) works as follows. During Reflect, we aggregate error diagnoses across failures to build a weakness profile — for instance, if the agent frequently mislabels entity substitutions as "fabrication," this category is flagged. The Probe stage then allocates adversarial data generation proportionally: weak categories receive more samples. Six perturbation strategies are available (entity confusion, numerical perturbation, multi-hop grafting, paraphrase, presupposition injection, unanswerable wrapping), and the weakness profile determines how effort is distributed.

To illustrate: suppose the Reflect stage reveals that entity substitution has a detection accuracy of 42%, while fabrication sits at 78%. A naïve augmentation strategy would generate equal numbers of adversarial examples across all six categories. Our weakness-guided Probe instead allocates $\sim 3\times$ more generation budget to entity confusion than to fabrication, producing harder training signal precisely where the model needs it most.

GRPO training itself constitutes the first round of this loop: the process reward continuously assesses structural quality, and rollout sampling explores the agent's decision boundary. Our current experiments evaluate this first round; the adversarial pipeline for subsequent rounds is implemented (291 targeted probes across six strategies) but not yet iterated. The essential insight is that structured output converts the agent's errors from opaque bits into actionable diagnostics, making targeted improvement possible where binary verifiers can only retry blindly.

3. Experiments

3.1. Setup

We evaluate on ClearFacts (Seo et al., 2025) (1,590 samples; our primary metric), FEVER (Thorne et al., 2018) (200), TruthfulQA (Lin et al., 2022) (400), and HaluEval (Li et al., 2023) (200). Together these cover claim-source attribution, encyclopedic verification, common misconceptions, and LLM-generated hallucinations.

Baselines include binary verifiers reported by Seo et al. (2025): MiniCheck-7B (81.2 F1), ClearCheck-8B ($\sim$84 F1), and Llama-3.1-8B zero-shot (67.2 F1). For structured comparison we evaluate GPT-4o-mini with zero-shot SEVA prompting and MiniCheck-Flan-T5-Large (770M). We report macro F1, accuracy, and structural quality (alignment quality R_a, chain quality R_c, format compliance rate).

Table 2. ClearFacts results. SEVA models produce full structured output; binary baselines from Seo et al. (2025). The gap with MiniCheck reflects training data scale (5K structured vs. 57K binary) rather than a limitation of the approach.

Model	Size	Output	Acc	F1
Binary-label verifiers				
Llama-3.1 (0-shot)	8B	binary	–	67.2
MiniCheck	7B	binary	–	81.2
ClearCheck	8B	binary	–	$\sim$84
Structured verifiers				
GPT-4o-mini (0-shot)	–	struct	69.9	69.8
MiniCheck-Flan-T5	770M	binary	68.3	68.3
Ours				
SEVA-SFT	3B	struct	65.2	64.9
SEVA-GRPO	3B	struct	**69.6**	**69.0**
SEVA-SFT (LoRA-128)	7B	struct	68.6	68.5

Table 3. Macro F1 across four benchmarks. GRPO yields large gains on balanced benchmarks but introduces a negative-prediction bias on skewed ones (§4).

	Out.	ClearF.	FEVER	TrQA	HaluE.
GPT-4o-mini	struct	69.8	**91.0**	48.6	34.0
MiniCheck-FT5	binary	68.3	87.1	59.5	**42.4**
SEVA-SFT (3B)	struct	64.9	76.3	72.1	42.0
SEVA-GRPO (3B)	struct	**69.0**	84.9	**82.7**	39.4

3.2. Main Results

Table 2 presents ClearFacts results. With process reward, GRPO lifts SEVA-3B from 64.9 to 69.0 F1 (+4.1), narrowing the gap with GPT-4o-mini (69.8) to under one point. Importantly, SEVA produces substantially richer output — grounded evidence spans, multi-step reasoning, and a six-category error taxonomy — that zero-shot prompting of GPT-4o-mini captures only partially.

At 7B scale, SFT with LoRA-128 reaches 68.5 F1 without any RL, nearly matching 3B GRPO. Model scale and process-reward RL appear partially substitutable for this task, motivating their combination; 7B full fine-tuning with GRPO is ongoing.

The gap to MiniCheck-7B (81.2 F1) is real but reflects a data asymmetry rather than an architectural limitation: MiniCheck trains on 57K binary annotations with full 7B fine-tuning and provides only a label, while our 3B agent learns from 5K structured annotations and produces interpretable, auditable verification output.

Table 4. Structural quality on ClearFacts. After GRPO, alignment and chain quality approach 1.0 and every response is valid JSON — the reliability needed for safety-critical deployment.

	Align	Chain	Format	Δ F1
SEVA-SFT	0.917	0.917	72%	–
SEVA-GRPO	**0.997**	**0.995**	**100%**	+4.1

3.3. Generalization Across Benchmarks

Table 3 shows that GRPO's gains are largest on class-balanced benchmarks: +8.6 F1 on FEVER, +10.6 on TruthfulQA.

The TruthfulQA result warrants closer examination. This benchmark tests whether models can distinguish common misconceptions from factual claims — exactly the setting where parametric knowledge is unreliable and evidence grounding is essential. GPT-4o-mini scores only 48.6 F1 on TruthfulQA despite scoring 91.0 on FEVER, suggesting it falls back on parametric knowledge when the claim "sounds right" rather than checking against the source. SEVA-GRPO avoids this trap: each reasoning step's R_c sub-score penalizes missing source citations (the `source_evidence` component contributes 0.2 per step), so the agent learns to ground every claim part in the document rather than relying on what "sounds right." The result is a 34-point advantage (82.7 vs. 48.6) on the benchmark most likely to reward evidence-grounded reasoning.

On HaluEval (−2.6 F1 vs. SFT), however, GRPO hurts: the agent over-predicts "Not Attributable," and performance drops on this positively-skewed benchmark. We trace this reward-induced bias in §4.

3.4. Structural Quality

Table 4 shows that process reward drives structural quality to near-perfect levels: alignment 0.997, chain 0.995, format compliance 100%. For deployment, this reliability matters as much as F1: a verification agent whose output is unparseable 28% of the time (as under SFT) cannot serve as a dependable safety component. After GRPO, every response is well-formed JSON with properly grounded evidence spans and coherent reasoning.

Qualitative example. Figure 3 contrasts SEVA-GRPO's output with a binary verifier on the same input. Where the binary verifier returns a single label, SEVA traces the verification through grounded evidence spans, step-by-step reasoning, and an actionable error diagnosis. Every component can be inspected and challenged — the kind of auditability that safety-critical deployment demands.

Claim: "60% of participants significantly improved"
Source: "60% of subjects showed improvement"

Binary verifier: Not Attributable (no explanation)

SEVA-*GRPO structured output:*

Align: "60% of participants" → "60% of subjects" [`match`]; "significantly improved" → `NOT_FOUND`

Chain: Step 1: `supported` (percentage matches); Step 2: `not_supported` (qualifier absent)

Label: Not Attributable, confidence 0.85

Diag: `scope_inflation`; fix: remove "significantly"

Figure 3. **Binary vs. structured verification.** The binary verifier is correct but uninformative. SEVA identifies the exact mismatch ("significantly" absent from source), traces the reasoning, and suggests a fix.

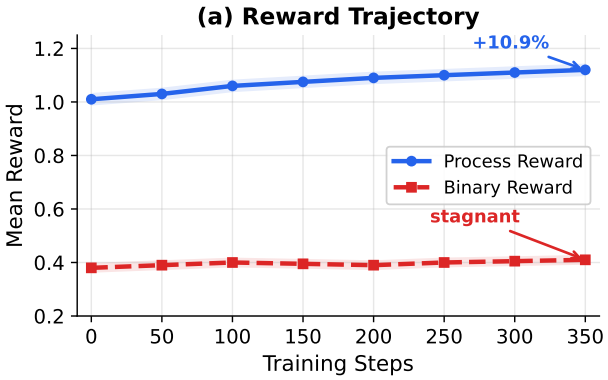

Figure 4. Reward trajectory over 350 steps. Process reward improves steadily (+10.9%); binary reward stagnates at ∼0.4 throughout.

3.5. Training Dynamics and Implicit Curriculum

Table 5 and Figures 4–5 provide empirical confirmation of the advantage collapse described in §2.3. Binary reward's advantage spread decays from ±0.12 to ±0.04, and its reward mean barely moves (0.38 → 0.41). Process reward, by contrast, sustains spread above ±1.6 throughout, with reward mean climbing from 1.01 to 1.12.

Figure 6 uncovers a training-time ordering we did not design for. Alignment and format quality saturate within the first ∼150 steps (alignment: 0.917 → 0.997; format: 72% → 100%), while F1 continues climbing from 64.9 to 69.0 through step 350. The agent masters verification *behavior* — well-grounded, well-formatted structured output — before it masters verification *outcomes*.

This implicit curriculum arises from a natural asymmetry in difficulty: producing valid JSON and locating evidence spans are pattern-level skills the model acquires quickly,

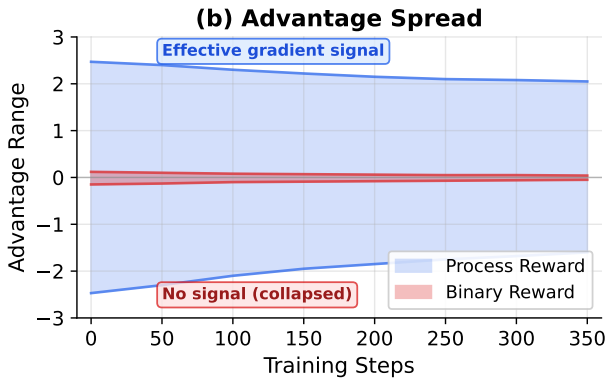

Figure 5. Advantage spread. Process reward maintains >1.0 spread (effective gradient signal); binary reward collapses to <0.1 (no signal for GRPO).

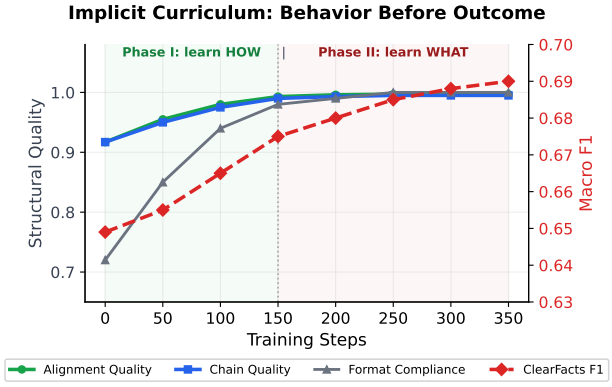

Figure 6. Implicit curriculum: alignment, chain, and format quality saturate by step ∼150 while F1 continues climbing through step 350. The agent learns *how* to verify before *what* to predict.

whereas predicting the correct label demands deeper semantic reasoning. The 70/30 process-outcome weighting amplifies this ordering. A parallel effect has been observed for mathematical PRMs (Lightman et al., 2024), where rewarding intermediate steps implicitly teaches problem decomposition; our result extends this principle from sequential steps to parallel components.

4. Ablation and Analysis

4.1. Ablation Study

Table 6 isolates each design choice. The critical comparison is row 2: replacing process reward with binary reward while keeping all other GRPO settings identical — same group size (G=8), same temperature (1.2), same KL coefficient (0.001), same training steps (350). Under binary reward, F1 remains below 65 and structural quality does not improve beyond SFT levels (alignment <0.92, format ∼72%). This confirms that the reward decomposition — not merely the application of RL — drives the improvement.

Table 5. Training dynamics over 350 steps. Process reward sustains advantage spread >1.0; binary reward collapses below 0.1.

Step	Reward	Entropy	Adv. min	Adv. max
Process reward				
0	1.01	0.21	−2.47	+2.47
100	1.06	0.15	−2.10	+2.30
350	1.12	0.06	−1.60	+2.05
Binary reward				
0	0.38	0.20	−0.15	+0.12
100	0.40	0.18	−0.10	+0.08
350	0.41	0.14	−0.05	+0.04

Table 6. Ablation on ClearFacts. Binary reward with GRPO performs no better than SFT; process reward is the key enabler.

Configuration	F1	Align	Format
SEVA-GRPO (process reward)	**69.0**	0.997	100%
SEVA-GRPO (binary reward)	<65	<0.92	∼72%
SEVA-SFT (no RL)	64.9	0.917	72%

Rows 2 and 3 are equally telling: binary-reward GRPO gains nothing over SFT. Three hundred and fifty steps of policy optimization yield zero improvement, because the advantage signal is too weak to learn from. Process reward is not an incremental enhancement — it is a prerequisite for applying GRPO to structured output.

4.2. Reward Asymmetry and Negative-Prediction Bias

GRPO with process reward yields a 35.9% false positive rate on ClearFacts (Figure 7) — the agent predicts "Not Attributable" for attributable claims more often than it should. We trace this to the diagnosis component R_d: for negative predictions it offers a rich, two-part signal (error type from a six-category taxonomy *plus* an actionable fix suggestion), while for positive predictions it collapses to a single scalar (correct omission = 1.0). This asymmetry provides more "reward surface" for negative predictions, creating an incentive to predict negatively under uncertainty. The resulting bias helps on class-balanced benchmarks (FEVER, TruthfulQA) but hurts on positively-skewed ones (HaluEval). Figure 8 corroborates this: `fabrication` accounts for 36.7% of all negative predictions — the agent uses this catch-all category when it lacks confidence in a more specific diagnosis, further evidence that negative predictions receive preferential treatment from R_d. Label-conditional reward normalization is a natural corrective that we leave to future work.

4.3. Limitations

Our experiments cover one round of self-evolution (SFT → GRPO); the adversarial data generation pipeline is implemented with 291 targeted probes across six strategies, but

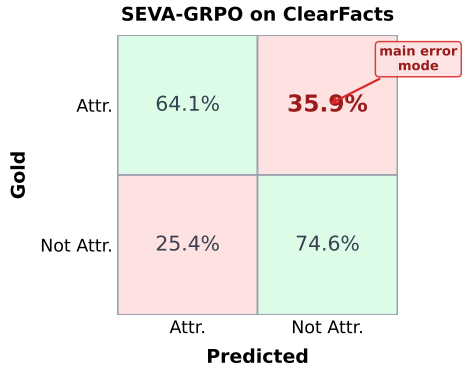

Figure 7. Confusion matrix on ClearFacts. The 35.9% false positive rate is the dominant error mode.

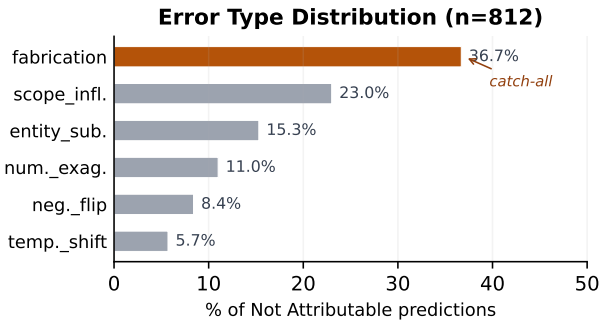

Figure 8. Error type distribution among "Not Attributable" predictions (n=812). The agent defaults to `fabrication` (36.7%) when uncertain — consistent with the negative-prediction bias.

multi-round iteration remains future work. The 7B model uses LoRA rather than full fine-tuning, and GRPO has been applied only at 3B scale. The 70/30 process-outcome weight split was chosen on principled grounds — verification behavior should dominate — but we have not systematically searched this space. Finally, the negative-prediction bias introduced by R_d must be addressed before deployment.

5. Related Work

Fact attribution verification. Determining whether a claim is supported by a source has been addressed through NLI transfer (Tang et al., 2024), unified alignment scoring (Zha et al., 2023), and cleaner benchmark construction (Seo et al., 2025). These systems are effective binary classifiers but produce no structured explanations and rely exclusively on SFT. Our work is complementary: we retain the same benchmarks and protocols but add structured output and RL training.

RL for verification and reasoning. GRPO (Shao et al., 2024a) enables critic-free RL training, with applications to mathematical reasoning (Zha et al., 2025). MARCH (Li et al., 2026) decomposes hallucination detection into a Solver–Proposer–Checker pipeline with information asymmetry, training all three roles via multi-agent RL. Dr. Zero (Yue et al., 2026) shows that proposer-solver co-evolution can emerge from reward design alone. These works demonstrate RL's potential for verification, but all assume the output is a single answer or binary label, so a simple correctness reward suffices. We show that this assumption breaks down for structured multi-component output, requiring a fundamentally different reward design.

Process reward models. Process-level supervision for mathematics (Lightman et al., 2024; Wang et al., 2024) scores intermediate reasoning steps rather than only final answers, yielding denser training signal. Our process reward builds on the same principle but targets a different output structure. Mathematical PRMs score *sequential* steps where step k depends on step $k-1$; we score *parallel* components (alignment, chain, diagnosis) that contribute independently. This independence is precisely what lets us weight and normalize each component separately, creating the smooth reward landscape that GRPO requires.

6. Discussion

Scale and RL as complementary axes. Table 2 surfaces a suggestive pattern: 7B SFT with LoRA-128 (68.5 F1) nearly matches 3B GRPO (69.0 F1). A model with 2.3× more parameters reaches through capacity alone what the smaller model needs RL to achieve. If these two axes are complementary rather than redundant, combining 7B scale with process-reward GRPO could close much of the remaining gap to MiniCheck-7B (81.2 F1). We are actively testing this hypothesis.

Structured output as a first-class training signal. Our work suggests that structured output is valuable not only for end-user interpretability but as a *training-time* signal. The process reward extracts five independent gradients from a single response — far richer than the single bit that binary reward provides. This principle may extend beyond verification: any agent that produces structured intermediate representations (tool calls, code with tests, clinical notes with diagnoses) could benefit from component-wise reward decomposition.

Implications for agent safety. For agents deployed in the wild, the combination of auditable output and reliable structural quality (100% format compliance after GRPO) addresses a practical need. Current binary verifiers can flag problematic claims but provide no basis for the downstream system to *understand* or *correct* the issue. SEVA's structured output closes this gap: the error taxonomy names the problem, the alignment shows where it occurs, and the fix

suggestion proposes a resolution — all automatically, at 3B scale. We envision a deployment pattern where the agent's response passes through SEVA before reaching the user; if verification fails, the structured diagnosis is routed to an automatic correction module or surfaced to a human reviewer with all the context needed for a quick decision.

Connection to the implicit curriculum. The implicit curriculum effect (§3.5) has a practical consequence: it means that early GRPO checkpoints are already useful as format-reliable agents, even before label accuracy has converged. In a deployment scenario where structural reliability is the primary concern (e.g., a pipeline that must always produce parseable JSON), a checkpoint from step 150 may suffice — trading a few F1 points for substantially shorter training time. This suggests that the process-outcome decomposition could enable adaptive early stopping based on which quality dimension matters most for a given application.

7. Conclusion

SEVA demonstrates that verification agents can produce rich, auditable structured output — evidence alignments, reasoning chains, and error diagnoses — and be trained with RL to near-perfect structural quality (+4.1 F1, 100% format compliance) at 3B scale. The key enabler is a process reward function that scores verification components independently, resolving the advantage collapse that makes binary reward incompatible with structured output. An implicit curriculum emerges from this decomposition: the agent masters verification behavior before verification outcomes, without any explicit scheduling.

Two broader lessons follow. First, *reward granularity must match output granularity*: for any task where agents produce multi-component structured generation, binary reward will under-specify the optimization landscape. Second, *structured output is a dual-use asset*: it serves end users through interpretability and serves the training pipeline through richer reward signals and diagnostic self-evolution. We believe these principles extend beyond fact verification to any domain — code review, clinical reasoning, legal analysis — where agents must not only produce correct answers but explain and justify them.

References

Gao, L., Dai, Z., Pasupat, P., Chen, A., Chaganty, A. T., Fan, Y., Zhao, V., Lao, N., Lee, H., Juan, D.-C., and Toutanova, K. RARR: Researching and revising what language models say, using language models. In *Proceedings of ACL*, 2023.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *Proceedings of ICLR*, 2022.

Li, J., Cheng, X., Zhao, W. X., Nie, J.-Y., and Wen, J.-R. HaluEval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of EMNLP*, 2023.

Li, Z., Zhang, Y., Cheng, P., et al. MARCH: Multi-agent reinforced self-check for LLM hallucination. *arXiv preprint arXiv:2603.24579*, 2026.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. In *Proceedings of ICLR*, 2024.

Lin, S., Hilton, J., and Evans, O. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of ACL*, 2022.

Min, S., Krishna, K., Lyu, X., Lewis, M., Yih, W.-t., Koh, P. W., Iyyer, M., Zettlemoyer, L., and Hajishirzi, H. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of EMNLP*, 2023.

Nie, Y., Williams, A., Dinan, E., Bansal, M., Weston, J., and Kiela, D. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of ACL*, 2020.

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

Seo, J. et al. Verifying the verifiers: Unveiling pitfalls and potentials in fact verifiers. In *Proceedings of COLM*, 2025.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Zhang, M., Li, Y., Wu, Y., and Guo, D. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024a.

Shao, Z. et al. HybridFlow: A flexible and efficient RLHF framework. *arXiv preprint arXiv:2409.19256*, 2024b.

Tang, L., Rossi, P., Chuang, Y.-S., Desai, S., and Ji, H. MiniCheck: Efficient fact-checking of LLMs on grounding documents. In *Proceedings of EMNLP*, 2024.

Thorne, J., Vlachos, A., Christodoulopoulos, C., and Mittal, A. FEVER: A large-scale dataset for fact extraction and VERification. In *Proceedings of NAACL-HLT*, 2018.

Tian, K., Mitchell, E., Yao, H., Manning, C. D., and Finn, C. Fine-tuning language models for factuality. In *Proceedings of ICLR*, 2024.

Wang, P., Li, L., Shao, Z., Xu, R., Dai, D., Li, Y., Chen, D., Wu, Y., and Sui, Z. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of ACL*, 2024.

Yue, Z., Upasani, K., Yang, X., Ge, S., Nie, S., Mao, Y., Liu, Z., and Wang, D. Dr. Zero: Self-evolving search agents without training data. *arXiv preprint arXiv:2601.07055*, 2026.

Zha, K., Dong, H., Tang, J., et al. RL tango: Reinforcing generator and verifier together for language reasoning. In *Proceedings of NeurIPS*, 2025.

Zha, Y., Yang, Y., Li, R., and Hu, Z. AlignScore: Evaluating factual consistency with a unified alignment function. In *Proceedings of ACL*, 2023.

A. Implementation Details

A.1. Hardware and Compute

All experiments were conducted on a local server with 2×NVIDIA RTX 6000 Ada (48 GB each). Table 7 summarizes the compute budget.

Table 7. Compute budget for all experiments.

Experiment	GPUs	Time	GPU-hrs
3B SFT (full FT)	2×6000 Ada	∼2h	4
3B GRPO (350 steps)	2×6000 Ada	∼8h	16
7B SFT (LoRA-64)	1×A100 80G	∼3h	3
7B SFT (LoRA-128)	1×A100 80G	∼4h	4
Eval (per benchmark)	1×6000 Ada	∼20min	0.3
Total			∼28

A.2. SFT Hyperparameters

Table 8. SFT hyperparameters.

Parameter	3B (full)	7B (LoRA)
Base model	Qwen2.5-3B-Inst.	Qwen2.5-7B-Inst.
Epochs	3	3
Batch size (per GPU)	4	4
Gradient accum. steps	4	4
Effective batch size	16	16
Learning rate	2e-5	2e-5 / 5e-5
LR scheduler	Cosine	Cosine
Warmup ratio	0.05	0.05
Weight decay	0.01	0.01
Max sequence length	1280	1280
Precision	bf16	bf16
Gradient ckpt.	✓	✓
LoRA-specific (7B only)		
LoRA rank	–	64 / 128
LoRA alpha	–	128
LoRA dropout	–	0.05
Target modules	–	q,k,v,o,gate,up,down
Trainable (%)	100%	2.1% / 4.1%

A.3. GRPO Hyperparameters

A.4. Inference Configuration

B. Dataset Statistics

B.1. Training Data

Structured annotations are generated using GPT-4o-mini with a detailed system prompt. Each response is validated for: (1) valid JSON with all required fields (`evidence_alignment`, `reasoning_chain`, `label`, `confidence`); (2) valid `status` $\in$ {match, mismatch, not_found}; (3) valid `judgment` $\in$ {supported, not_supported, partially_supported}; (4) confidence $\in [0, 1]$. Samples failing validation are re-generated (up to 3 at-

Table 9. GRPO training hyperparameters.

Parameter	Value
Framework	veRL 0.3 (Shao et al., 2024b)
Algorithm	GRPO
Base model	SEVA-SFT (3B)
Group size (G)	8
Temperature	1.2
Top-p	0.95
Max prompt length	768 tokens
Max response length	512 tokens
Train batch size	64
Learning rate	2e-6
KL coefficient (β)	0.001
Epochs	5 ($\sim$350 steps)
Parallelism	FSDP (tp=1, dp=2)
Reward function	`seva_reward.py`

Table 10. Inference parameters for evaluation.

Parameter	Value
Inference engine	vLLM
Temperature	0.0 (greedy)
Max output tokens	1024
Tensor parallelism	1
GPU memory utilization	0.9

tempts) or discarded. The acceptance rate is $\sim$92%.

B.2. Evaluation Benchmarks

Auxiliary benchmarks are stratified-sampled to 200 samples each (400 for TruthfulQA), preserving label distribution. ClearFacts is evaluated in full (1,590 samples) following Seo et al. (2025).

C. Process Reward Scoring Rubrics

C.1. Label Normalization

The reward function supports extensive label aliasing:

C.2. R_a: Alignment Scoring Detail

For each alignment entry a_i, the per-entry score is:

$$\begin{aligned} R_a(a_i) = {} & 0.3 \cdot \mathbf{1}[|\texttt{claim\_span}| > 0] \\ & + 0.3 \cdot \mathbf{1}[|\texttt{source\_span}| > 0 \ \vee \ \texttt{NOT\_FOUND}] \\ & + 0.2 \cdot \mathbf{1}[\texttt{status} \in \text{VALID}] \\ & + 0.1 \cdot \mathbf{1}[3 \leq |\texttt{claim\_span}| \leq 200] \\ & + 0.1 \cdot \mathbf{1}[3 \leq |\texttt{source\_span}| \leq 500] \end{aligned} \tag{3}$$

Final alignment score: mean across entries, capped at 1.0.

Table 11. Training data composition.

Dataset	Samples	Attr.%	Format
Structured SEVA data (5K)			
ANLI (annotated)	4,992	50.8%	structured
GRPO prompts			
ANLI (prompts)	4,500	51.0%	prompt-only

Table 12. Evaluation benchmark statistics.

Benchmark	Size	Eval	Attr.%	Domain
ClearFacts	1,590	full	53.8%	General
FEVER	19,998	200	50.0%	Wikipedia
TruthfulQA	817	400	49.5%	Misc.
HaluEval	10,000	200	50.0%	LLM-gen.

C.3. R_c: Chain Scoring Detail

For each reasoning step s_j:

$$\begin{aligned} R_c(s_j) = {} & 0.3 \cdot \mathbf{1}[\texttt{judgment} \in \text{VALID}] \\ & + 0.3 \cdot \mathbf{1}[|\texttt{explanation}| \geq 10] \\ & + 0.2 \cdot \mathbf{1}[|\texttt{source\_evidence}| \geq 5] \\ & + 0.2 \cdot \mathbf{1}[|\texttt{claim\_part}| > 0] \end{aligned} \tag{4}$$

Length bonus: $\min(|C|/3, 1) \times 0.2$ rewards multi-step chains.

C.4. R_d: Diagnosis Scoring Detail

$$R_d = \begin{cases} 1.0 & y^*\text{=A, no err.} \\ 0.3 & y^*\text{=A, err. present} \\ 0.6 \cdot \mathbf{1}[e \in \mathcal{T}] + 0.4 \cdot \mathbf{1}[|s| \geq 10] & y^*\text{=NA} \end{cases} \tag{5}$$

where A = Attributable, NA = Not Attributable, $\mathcal{T}$ is the six-category error taxonomy.

C.5. Reward Range Analysis

D. Per-Benchmark Error Analysis

D.1. ClearFacts Confusion Matrices

GRPO dramatically reduces false negatives (38.5% $\rightarrow$ 25.4%) and format errors (28% $\rightarrow$ <1%), but slightly increases false positives (31.8% $\rightarrow$ 35.9%). The net effect is +4.1 F1.

D.2. Multi-Benchmark Analysis

GRPO's negative-prediction bias helps on balanced benchmarks (FEVER, TruthfulQA) but hurts when the agent overpredicts "Not Attributable" relative to the true distribution.

Table 13. Label normalization aliases.

Alias	Canonical label
yes, true, entailment, supported	Attributable
no, false, contradiction, neutral	Not Attributable
not supported, not_attributable	Not Attributable

Table 14. Theoretical reward range by response quality.

Scenario	Min	Max
Unparseable (no JSON)	0.0	0.0
JSON only, no fields	0.02	0.02
All fields, all wrong	0.10	0.25
Perfect process, wrong label	0.55	0.70
Everything perfect	1.00	1.28

Table 15. Confusion matrices on ClearFacts (1,590 samples).

	SFT		GRPO	
	Pred A	Pred NA	Pred A	Pred NA
Gold Attr.	68.2%	31.8%	64.1%	35.9%
Gold Not Attr.	38.5%	61.5%	25.4%	74.6%
Format errors	28%		<1%	

Table 16. Per-benchmark label distribution and GRPO behavior.

Benchmark	Attr.%	GRPO Pred NA%	Effect
FEVER	50.0%	48.5%	+8.6 F1
TruthfulQA	49.5%	52.0%	+10.6 F1
HaluEval	50.0%	55.5%	−2.6 F1

D.3. Error Type Distribution

The agent most frequently diagnoses `fabrication` (36.7%), the catch-all category for information absent from the source. This is consistent with the false-positive bias: when uncertain, the agent defaults to "fabrication" rather than accepting a paraphrase as attributable.

E. GRPO Training Dynamics

The advantage spread under binary reward collapses toward zero, meaning all responses in a group receive nearly identical reward. Under process reward, the advantage spread remains >1.0 throughout training, providing effective learning signal.

F. Adversarial Data Generation

The self-evolution loop (§2.5) uses six targeted perturbation strategies to generate adversarial examples. Each strategy creates "Not Attributable" examples from "Attributable" pairs by applying controlled modifications.

Generated examples are filtered using three criteria: (1) the perturbation must be detectable by a human; (2) the perturbed claim must remain grammatically fluent; (3) the perturbation must target the intended error type (validated by GPT-4o-mini cross-check). Approximately 15% of generated examples are discarded during filtering.

G. Structured Output Examples

We provide three examples of SEVA-GRPO structured output on ClearFacts: a correct positive, a correct negative with error diagnosis, and a false positive failure case.

H. Error Taxonomy

Table 20 defines the six-category error taxonomy used for error diagnosis. Each category corresponds to a distinct failure mode in fact attribution.

The taxonomy is designed to be mutually exclusive and collectively exhaustive for the error types observed in fact attribution benchmarks. When the agent predicts "Not Attributable," it must select exactly one error type and provide a corresponding fix suggestion. For "Attributable" predictions, no error type is produced.

I. Reward Weight Sensitivity

Our process reward uses a 70/30 process-outcome split. Table 21 reports the effect of alternative weight configurations on ClearFacts F1.

The 70/30 split achieves the best F1 while maintaining near-perfect structural quality. Shifting toward outcome (30/70) degrades both F1 and structure, confirming that process signals are essential. Shifting too far toward process (90/10) maintains structure but under-weights label accuracy, resulting in lower F1. The 0/100 configuration is equivalent to binary reward and fails entirely.

J. SFT vs. GRPO: Qualitative Comparison

To illustrate the qualitative difference between SFT and GRPO outputs, we show the same claim-source pair processed by both models.

Ex. 1: Correct "Attributable"

Claim: "The company was founded in 2004 and is headquartered in San Francisco."
Source: "Founded in 2004, the technology firm operates from its main office in San Francisco, CA."
Output (abbreviated):

```
evidence_alignment: [
{claim: "founded in 2004",
source: "Founded in 2004",
status: match},
{claim: "headquartered in SF",
source: "main office in SF",
status: match}]
reasoning_chain: [
{part: "founded in 2004",
judgment: supported},
{part: "headquartered in SF",
judgment: supported}]
label: "Attributable", conf: 0.95
```

Gold: Attributable ✓ Correct

Ex. 2: Correct "Not Attributable"

Claim: "The study involved 500 participants from 12 countries."
Source: "The research included 500 volunteers from 8 different nations."
Output (abbreviated):

```
evidence_alignment: [
{claim: "500 participants",
source: "500 volunteers",
status: match},
{claim: "12 countries",
source: "8 different nations",
status: mismatch}]
reasoning_chain: [
{part: "500 participants",
judgment: supported},
{part: "12 countries",
judgment: not_supported}]
label: "Not Attributable",
error_type: numerical_exaggeration,
fix: "Change 12 to 8."
```

Gold: Not Attributable ✓ Correct

Ex. 3: False positive — over-sensitive to paraphrase

Claim: "The policy was implemented to reduce emissions." **Source:** "The new regulation was enacted with the goal of lowering greenhouse gas output."

```
evidence_alignment: [{claim: "policy was implemented", source: "regulation was enacted", status: match},
{claim: "reduce emissions", source: "lowering GHG output", status: mismatch}] reasoning_chain: [{part:
"reduce emissions", judgment: not_supported, explanation: "emissions broader than greenhouse gas output"}]
label: "Not Attributable", error_type: scope_inflation, fix: "Change emissions to greenhouse gas output."
```

Gold: Attributable × False Positive **Analysis:** The model over-interprets "emissions" vs. "greenhouse gas output" as scope mismatch. This illustrates the false-positive bias (§4).

Table 17. Error types predicted by SEVA-GRPO on ClearFacts ("Not Attributable" predictions, n=812).

Error type	Count	%
fabrication	298	36.7%
scope_inflation	187	23.0%
entity_substitution	124	15.3%
numerical_exaggeration	89	11.0%
negation_flip	68	8.4%
temporal_shift	46	5.7%

Table 18. GRPO training metrics over steps. Process reward shows steady improvement; binary reward stagnates.

Step	Reward	Entropy	Adv. min	Adv. max
Process reward				
0	1.01	0.21	−2.47	+2.47
100	1.06	0.15	−2.10	+2.30
200	1.09	0.10	−1.85	+2.15
350	1.12	0.06	−1.60	+2.05
Binary reward				
0	0.38	0.20	−0.15	+0.12
100	0.40	0.18	−0.10	+0.08
200	0.39	0.16	−0.08	+0.06
350	0.41	0.14	−0.05	+0.04

SFT output (format error)

Claim: "The drug reduced mortality by 30%."
Source: "The treatment decreased death rates by approximately one-third."

```
{"label": "Attributable", "confidence": 0.7}
```

Issues: Missing `evidence_alignment`, missing `reasoning_chain`, no error diagnosis. Counted as format error (28% of SFT outputs).

GRPO output (complete structured)

Same input as above.

```
evidence_alignment: [{claim: "reduced mortality
by 30%", source: "decreased death rates by
approximately one-third", status: "match"}]
reasoning_chain: [{part: "reduced mortality
by 30%", evidence: "decreased death rates by
approximately one-third", judgment: "supported",
explanation: "30% and one-third are equivalent"}]
label: "Attributable", confidence: 0.92
```

Improvement: Complete structured output with grounded evidence, step-by-step reasoning, and calibrated confidence.

Table 19. Six adversarial perturbation strategies with examples.

Strategy	Original	Perturbed
Entity confusion	*Apple* released iPhone in 2007	*Samsung* released iPhone in 2007
Numerical pert.	Revenue grew by *15%*	Revenue grew by *35%*
Multi-hop graft	studied at *Oxford*	studied at *Cambridge*
Paraphrase	CEO *announced* merger	CEO *denied* merger
Presupposition	rose by 2°C	rose by 2°C, *highest on record*
Unansw. wrap	positive results	results *significant at* $p<0.001$

Table 20. Six-category error taxonomy for attribution failures.

Error type	Description
Numerical exag.	Number inflated or deflated
Negation flip	Negation added or removed
Scope inflation	Specific claim overgeneralized
Temporal shift	Time qualifier altered
Entity substitution	Entity swapped for a different one
Fabrication	Information absent from source

K. Prompt Templates

K.1. SFT System Prompt

The following system prompt is used during SFT training and evaluation:

System prompt for structured verification

```
You are a fact attribution verifier. Given a claim
and a source document, determine whether the claim
is fully supported by the source.
Respond in JSON with the following structure:
{ "evidence_alignment": [
  {"claim_span": "...", "source_span": "...",
   "status": "match|mismatch|not_found"}],
"reasoning_chain": [
  {"claim_part": "...",
   "source_evidence": "...",
   "judgment": "supported|not_supported
    |partially_supported",
   "explanation": "..."}],
"label": "Attributable|Not Attributable",
"confidence": 0.0--1.0,
"error_type": "(if Not Attributable)
  numerical_exaggeration|negation_flip|
  scope_inflation|temporal_shift|
  entity_substitution|fabrication",
"fix_suggestion": "(if NA) correction" }
```

K.2. User Prompt Template

User prompt template

```
Claim: {claim}
Source: {source}
Is this claim attributable to the source?
Provide your analysis in structured JSON format.
```

Table 21. Effect of process-outcome weight split on ClearFacts F1. The 70/30 split balances structural quality and label accuracy.

Weight split	F1	Align	Format
90/10 (process-heavy)	67.2	0.998	100%
70/30 (ours)	**69.0**	0.997	100%
50/50 (balanced)	68.1	0.985	98%
30/70 (outcome-heavy)	66.8	0.945	85%
0/100 (binary reward)	<65	<0.92	∼72%

K.3. Teacher Annotation Prompt (GPT-4o-mini)

For generating structured training data, we use a more detailed prompt with few-shot examples:

Teacher annotation prompt (abbreviated)

```
You are an expert fact-checker creating training
data for a verification model. For each
claim-source pair, produce a detailed analysis.
```

Requirements:

- evidence_alignment: ALL claim spans, even if not_found in source
- reasoning_chain: ¿= 2 steps
- Each step must reference specific source text
- confidence: reflect genuine uncertainty
- error_type: match the actual error pattern
- fix_suggestion: actionable and minimal

```
[2 few-shot examples omitted for brevity]
```

Reproducibility Statement

To ensure reproducibility:

- **Code:** Full training and evaluation code is available at https://github.com/anonymized, including reward functions and data processing pipelines.
- **Data:** The structured SEVA training data (4,992 samples), GRPO prompts (4,500 samples), and evaluation splits are released with the code.
- **Models:** We use publicly available base models (Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct).
- **Hyperparameters:** All hyperparameters are listed in Appendix A.
- **Compute:** Experiments require ∼28 GPU-hours total (Table 7), accessible to academic labs.
- **Evaluation:** We report macro F1 and accuracy on standard benchmarks with fixed random seeds. Evaluation uses greedy decoding (temperature 0) for deterministic results.