Correctable Fork Tokens: Verifier-Anchored Selective Credit Assignment for Tool-Integrated RLVR

Shuqi Yin
Independent Researcher
smartshuqiyin@gmail.com

Abstract

In tool-integrated reinforcement learning with verifiable rewards (RLVR), sequence-level verifier signals are issued for trajectories whose outcomes are determined by a sparse subset of tokens—tool names, argument keys, and argument values. Standard RLVR assigns the same advantage to every token in a rollout, diluting credit at precisely the positions that matter. We propose Correctable Fork Tokens (CFT), which decomposes selective credit assignment into two subproblems: *where* to update, and *how much* to update at those positions. To this end, CFT introduces a training-only answer-conditioned branch sharing parameters with the student policy. The key insight is that conditioning on the ground-truth answer concentrates the model's distribution at decision-fork positions, providing hindsight localization without additional parameter capacity or imitation targets. Critically, this benefit requires accurate answer information: a larger-capacity teacher without the answer fails to replicate these gains, confirming that privileged answer information—not teacher capacity—is the active ingredient. CFT uses this entropy reduction (IG_t) to identify correctable fork positions, then rescales token-level credit at those positions while preserving the verifier advantage as the sole signal for update direction. On BFCL v3/v4, CFT improves multi-turn accuracy by **4.2/6.5** pp and overall accuracy by **3.1/2.2** pp over GRPO; it further surpasses On-policy distillation—which employs a larger Expert Model as teacher—by **3.6/6.7** pp on multi-turn, confirming that answer-conditioned fork localization rather than teacher capacity drives the improvement. On Tau3, the average score improves by **8.3** pp. We additionally present a three-level structured verifier for tool-call settings; ablations confirm that finer-grained verifier feedback and selective token credit are complementary. Our reproducible implementation is available at `https://github.com/YSQ-boop/CFT`.

1 Introduction

In tool-integrated reasoning, verifiable rewards target trajectory-level outcomes, yet the tokens that determine these outcomes—tool names, argument keys, and argument values—form a sparse minority. Unlike math or code generation, tool-use trajectories contain many low-entropy format tokens alongside these structured decision tokens. Standard sequence-level RLVR assigns the same advantage to every position, diluting credit precisely where fine-grained assignment matters most. Existing approaches address only part of this: sequence-level RLVR preserves verifier-anchored optimization but provides coarse credit localization, while distillation methods provide hindsight signals but do not decompose *where to update* and *how much to update* as two distinct subproblems.

We propose Correctable Fork Tokens (CFT). The key insight is that conditioning on the ground-truth answer at *training time* concentrates the model's distribution at decision-fork positions, enabling hindsight localization without additional parameters or imitation targets. A training-only answer-conditioned branch identifies correctable decision forks via $IG_t = H_S(t) - H_T(t \mid \text{priv})$, which is

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

large precisely at positions uncertain for the student but resolved by the answer. Asymmetric KL then rescales credit magnitude at selected positions; the verifier remains the sole source of update direction.

CFT improves BFCL v3/v4 multi-turn accuracy by 4.2/6.5 pp and Tau3 average score by 8.3 pp over GRPO. CFT further surpasses On-policy distillation—which uses a larger Expert Model as teacher—by 3.6/6.7 pp on multi-turn, confirming that answer-conditioned fork localization, not teacher capacity, drives the improvement.

Our contributions are as follows:

1. We propose CFT, formalizing selective credit assignment for tool-call RLVR: a same-parameter answer-conditioned branch provides hindsight localization via IG_t; asymmetric KL rescales credit magnitude at selected positions; verifier-anchored update direction is preserved throughout.
2. We design a three-level structured verifier—tool-name alignment, argument-name supervision, and type-aware argument-value similarity—providing dense per-step feedback independently of CFT; Table 5 demonstrates that finer reward granularity and selective token credit are complementary.
3. We validate CFT on BFCL v3/v4 and Tau3 with ablations, answer-conditioning controls, and mechanism analysis confirming that gains stem from precise localization of decision-critical positions, not teacher capacity.

2 Related Work

Tool-integrated reasoning extends LLMs to external tools across single- and multi-turn settings [20, 19, 16, 18, 14, 17, 4, 29, 15]. Errors concentrate at structured decision positions—tool names, argument keys, argument values—but most approaches issue feedback at the step or sequence level without explicitly targeting which token positions should receive credit [8, 10]. RLVR avoids teacher-trajectory dependence [21, 30] but assigns uniform advantage to all tokens [13, 25]; distillation methods provide hindsight signals [1, 22, 31, 6] but target imitation rather than selective credit localization. Concurrent work [32] also explores structured tool rewards; CFT's verifier additionally uses type-aware argument-value matching and a union-style denominator $m+n-|M|$ handling missing and extra calls jointly.

Comparison with the most closely related work The most closely related work [13] identifies critical tokens via cross-rollout probability contrast, requiring contrastive pairs across trajectories. In comparison, CFT's IG_t requires only a single answer-conditioned forward pass; the signal comes from entropy reduction induced by answer conditioning rather than outcome contrast. [25] relaxes the KL penalty on identified tokens to encourage exploration; CFT concentrates credit magnitude on decision forks without modifying exploration, with the verifier advantage uniquely determining update direction. The key distinction: CFT relies on privileged answer conditioning available at training time, not on one-sided entropy statistics or cross-trajectory contrast.

Credit assignment and hindsight Classic credit assignment uses eligibility traces [23] or return decomposition [26]; recent process reward models [11] provide step-level credit for LLM reasoning—unlike PRMs, which require an independent reward model at inference time, CFT reweights token-level credit at training time using the verifier advantage as the sole direction signal. Hindsight experience replay [3] and privileged information learning [24] leverage training-time signals unavailable at test time. CFT similarly uses ground-truth answers to localize decision forks, but via entropy reduction rather than relabeling or distillation, preserving the verifier as the sole source of update direction. The population expectation of IG_t equals the conditional mutual information $I(Y_t; Z \mid C_t)$ between next token and privileged answer given context (Lemma B.5); CFT exploits this signal via a same-parameter forward pass rather than estimating population-level MI.

3 Method

Section 3.4 (CFT) is the primary contribution; Section 3.3 presents an independently applicable structured verifier, with complementarity demonstrated in Table 5.

3.1 Problem Setup

Given input question x, callable tool set $\mathcal{F}$, and an interaction environment, the language model generates a tool-integrated reasoning (TIR) trajectory y consisting of interleaved model reasoning, tool calls, and execution feedback. The model may call tools multiple times during multi-step reasoning and continue generating based on prior results.

During training we use a standard verifiable-reward setting. For each question x, student policy π_θ samples a rollout group $y^{(1)}, \ldots, y^{(G)} \sim \pi_\theta(\cdot \mid x)$. The verifier assigns sequence-level reward $R(x, y^{(i)})$ and the leave-one-out group-relative advantage [2] is $A^{(i)} = R(x, y^{(i)}) - \frac{1}{G-1} \sum_{j \neq i} R(x, y^{(j)})$. $A^{(i)}$ is the sole signal determining update direction: positive $A^{(i)}$ reinforces the trajectory, negative $A^{(i)}$ suppresses it.

We additionally construct a training-only, same-parameter, answer-conditioned branch as a teacher. The teacher shares parameters with the student and operates with stop-gradient throughout. The student observes question x; the teacher observes the same prefix plus privileged information priv, instantiated as the ground-truth per-step reference output: reference tool call(s) at a tool-call step and the reference `<response>` at a final-response step. For each sampled trajectory, the teacher computes answer-conditioned token-level distribution statistics; it does not serve as an imitation target and does not influence the direction of any reward or penalty. The teacher is used solely to identify which positions are likely correctable fork points and how much token-level credit they should receive. Sharing parameters with the student ensures that IG_t measures entropy reduction caused by the answer rather than capacity-induced distribution shifts: as the student policy evolves, the teacher distribution co-evolves identically, preventing the guidance signal from drifting off-policy.

This setup defines CFT's applicable scope: tool-integrated RLVR where ground-truth answers or equivalent privileged signals are accessible at training time. Under standard RLVR, token t receives advantage $A^{(i)}$, identical for every position; CFT approximates an ideal assignment concentrated on decision-critical fork positions while preserving the verifier as the sole source of update direction.

3.2 Rollout Format and Model Inputs

The system prompt requires the model to use `<think>`, `<tool_call>`, and `<response>` structured markers; `Tool List` specifies available tools at each step. The student input consists of the current question and compressed multi-turn history retaining `<user>`, `<response>`, and paired `<tool_call>`/`<tool_response>` entries while excluding `<think>` content; the teacher input additionally includes priv. Multi-step trajectories are decomposed into per-step instances. Detailed templates are in Appendix I.

3.3 Tool-Integrated Reward Design

The structured verifier in this section is designed independently of CFT's credit assignment, enabling isolated study of their interaction (Table 5). Relying solely on final-outcome reward is insufficient: it cannot distinguish invalid format, wrong tool selection, or incorrect argument generation. The three levels—tool-name alignment, argument-name supervision, and argument-value similarity—are captured by decomposing reward into a format component R_{format} and a correctness component R_{correct} [8, 10].

Let $P = \{p_1, \ldots, p_m\}$ and $\mathcal{G} = \{g_1, \ldots, g_n\}$ be predicted and reference call sequences, each consisting of a tool name and argument dictionary. We establish a greedy one-to-one matching M requiring tool-name agreement, with ties broken by argument-name overlap; duplicate calls are not double-counted.

For each matched pair $(i, j) \in M$, argument-name and argument-value agreement are evaluated separately. Argument-name agreement is defined as

$$r_{\text{argname}}(p_i, g_j) = \begin{cases} 1, & K(p_i) = K(g_j) = \varnothing, \\ \dfrac{|K(p_i) \cap K(g_j)|}{|K(p_i) \cup K(g_j)|}, & \text{otherwise}, \end{cases} \tag{1}$$

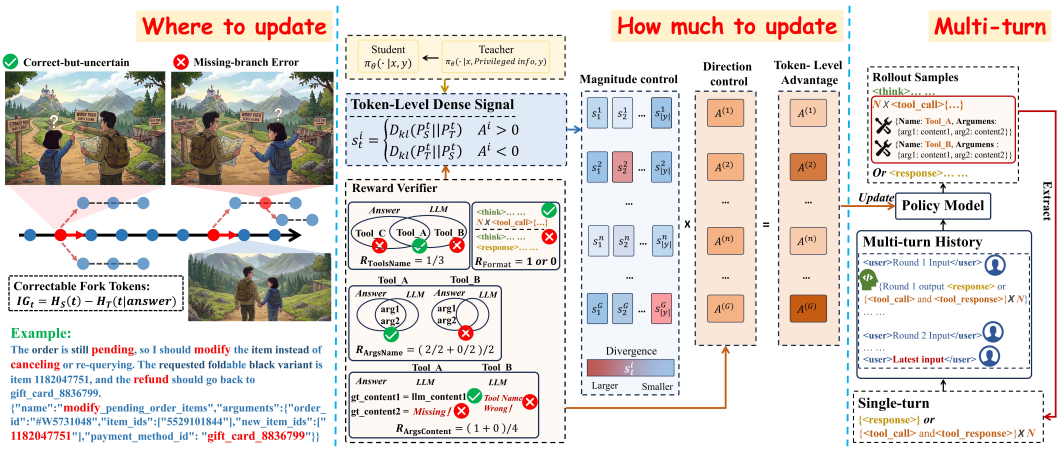


Figure 1: Overview of CFT. On multi-turn tool-interaction rollouts with compressed history, a training-only answer-conditioned teacher provides hindsight signals for selective credit assignment. `Where to update`: the information gap IG_t selects correctable fork tokens. `How much to update`: on the selected tokens, asymmetric KL rescales token-level credit magnitude, while the verifier-derived sequence-level advantage remains the only signal that determines update direction.

Argument values use a type-aware similarity function:

$$s_{\text{val}}(a,b) = \begin{cases} \text{ROUGE-L}_{F1}(a,b), & a,b \text{ are strings,} \\ \mathbf{1}[a=b], & a,b \text{ are numeric or boolean,} \\ \mathbf{1}[\text{canonical_str}(a) = \text{canonical_str}(b)], & \text{otherwise,} \end{cases} \tag{2}$$

where canonical_str serializes nested structures with sorted keys. String values use ROUGE-L F1 [12]. Based on this typed similarity, argument-value agreement is defined as

$$r_{\text{argvalue}}(p_i,g_j) = \begin{cases} 1, & K(p_i) = K(g_j) = \varnothing, \\ \dfrac{\sum_{k \in K(p_i) \cap K(g_j)} s_{\text{val}}(v(p_i,k), v(g_j,k))}{|K(p_i) \cup K(g_j)|}, & \text{otherwise.} \end{cases} \tag{3}$$

The per-pair fine reward score is $s_{ij}^{\text{fine}} = 1 + r_{\text{argname}}(p_i,g_j) + r_{\text{argvalue}}(p_i,g_j) \in [1,3]$ for any matched pair $(i,j) \in M$, where the constant 1 corresponds to a correct tool match; unmatched calls contribute 0 to the numerator of $R_{\text{Tools}}^{\text{fine}}$. To simultaneously penalize missing and extra calls, we apply union-style call-level normalization:

$$R_{\text{Tools}}^{\text{fine}} = \begin{cases} 3, & m = n = 0, \\ \dfrac{\sum_{(i,j)\in M} s_{ij}^{\text{fine}}}{m+n-|M|}, & \text{otherwise,} \end{cases} \qquad R_{\text{correct}}^{\text{fine}} = C \cdot \left(2 \cdot \frac{R_{\text{Tools}}^{\text{fine}}}{3} - 1\right) \in [-C, C]. \tag{4}$$

This normalization handles missing and extra calls in a single formula, where $C > 0$ is the correctness weight ($C = 3$ in experiments), and allows string arguments partial credit via ROUGE-L F1 [12].

The format component checks only whether the output follows a valid structured pattern:

$$R_{\text{format}} = \mathbf{1}\left[y \in \texttt{<think>}\left(\texttt{<response>} \mid \texttt{<tool_call>}^{N}\right)\right]. \tag{5}$$

This design explicitly separates syntactic validity from semantic correctness. For multi-step trajectories, $R(x,y^{(i)}) = \sum_s (R_{\text{format}}^{(s)} + R_{\text{correct}}^{(s)})$ summed over all tool-call steps. Full algorithmic details are in Appendix H.

3.4 Optimization Method

Based on the sequence-level advantage $A^{(i)}$ from Section 3.1 and the teacher's answer-conditioned distribution, CFT decomposes selective credit assignment into two subproblems: which positions to

update, and how much credit to assign at those positions. The update direction is always determined solely by $A^{(i)}$; the teacher provides decision-selective hindsight signal rather than a replacement optimization target. Figure 1 illustrates this decomposition.

Why IG_t works for tool-call trajectories Tool-call trajectories have a structural property that makes information-gap-based selection effective: format tokens have low entropy under both student and teacher ($IG_t \approx 0$); natural language tokens have high entropy under both ($IG_t \approx 0$); only structured decision tokens—tool names, argument keys, argument values—tend to be uncertain under the student but concentrated under the answer-conditioned teacher ($IG_t > 0$). High-IG_t positions therefore structurally correspond to correctable decision forks, not to format tokens or surface natural language. This structural argument is empirically confirmed in Section 4.5: 79% of IG_t-selected positions correspond to structured decision tokens.

Definition of correctable fork tokens Position t is a *correctable fork* if the student is uncertain ($H_S(t)$ is large) while the answer-conditioned teacher substantially reduces this uncertainty (IG_t is large), indicating the answer disambiguates a decision the student cannot resolve from context alone. Note that *structured decision tokens* (tool names, argument keys, values) are an objective structural category, while *correctable fork tokens* are the top-ρ IG_t-selected subset; the two overlap at 79% but are not identical.

At position t, let the student and teacher next-token distributions be $P_S^t(v) = \pi_\theta(v \mid x, y_{<t})$ and $P_T^t(v) = \pi_\theta(v \mid x, \text{priv}, y_{<t})$ respectively. We compute full-vocabulary entropies $H_S(t)$ and $H_T(t \mid \text{priv})$ and define the privileged-information-conditioned information gap as $IG_t = H_S(t) - H_T(t \mid \text{priv})$ (trajectory index omitted; full notation $IG_t^{(i)}$ in Algorithm 1). During training, we sort all positions in the batch by $IG_t^{(i)}$ globally and retain the top $\rho = 20\%$, yielding a binary mask

$$m_t^{(i)} = \mathbf{1}\left[IG_t^{(i)} \geq \tau_\rho^{\mathcal{B}}\right]. \tag{6}$$

This mask answers the question of where to update.

On the selected positions, we further compare student and teacher distributions in a local fork space. Full-vocabulary KL is dominated by low-probability tail tokens that inflate compute cost while providing only marginal additional signal at substantially higher memory cost; we therefore use a union top-k ($K = 20$ in all experiments) support as an engineering approximation—trading a small amount of tail-distribution fidelity for focused coverage of high-probability fork candidates and substantially reduced compute. We define the union top-k support as $\mathcal{V}_t^K = \text{TopK}(P_S^t) \cup \text{TopK}(P_T^t)$. We retain original probabilities on $\mathcal{V}_t^K$ and merge remaining probability mass into a tail bucket, yielding extended local distributions $\tilde{P}_S^t$ and $\tilde{P}_T^t$. The local fork salience is defined as

$$s_t^{(i)} = \begin{cases} D_{KL}(\tilde{P}_S^t \| \tilde{P}_T^t), & A^{(i)} > 0, \\ D_{KL}(\tilde{P}_T^t \| \tilde{P}_S^t), & A^{(i)} < 0, \\ 0, & A^{(i)} = 0. \end{cases} \tag{7}$$

This asymmetric design captures two types of decision fork errors: when $A^{(i)} > 0$, a large $D_{KL}(\tilde{P}_S^t \| \tilde{P}_T^t)$ indicates a correct-but-uncertain fork; when $A^{(i)} < 0$, a large $D_{KL}(\tilde{P}_T^t \| \tilde{P}_S^t)$ indicates a missing-branch error. We then normalize to a positive token-level multiplier $w_t^{(i)} = \frac{s_t^{(i)}}{\frac{1}{|S^{(i)}|}\sum_{j \in S^{(i)}} s_j^{(i)}}$, where $S^{(i)} = \{t : m_t^{(i)} = 1\}$, and construct the reweighted advantage $\bar{A}_t^{(i)} = A^{(i)} \cdot \text{clip}(w_t^{(i)}, 1-\epsilon_w, 1+\epsilon_w)$. Since the clipped multiplier is always positive, $\text{sign}(\bar{A}_t^{(i)}) = \text{sign}(A^{(i)})$; the teacher only amplifies or attenuates update magnitude at selected positions without altering the verifier-specified update direction.

Finally, the clipped surrogate objective operates only on the masked sampled tokens. Let $r_t^{(i)}(\theta) = \frac{\pi_\theta(y_t^{(i)} | x, y_{<t}^{(i)})}{\pi_{\theta_{\text{old}}}(y_t^{(i)} | x, y_{<t}^{(i)})}$, the training objective is

$$\mathcal{L}_{\text{CFT}}(\theta) = -\mathbb{E}\left[\frac{1}{\sum_{i=1}^{G}\sum_{t=1}^{|y^{(i)}|} m_t^{(i)}} \sum_{i=1}^{G}\sum_{t=1}^{|y^{(i)}|} m_t^{(i)} \min\left(r_t^{(i)}(\theta)\bar{A}_t^{(i)}, \text{clip}(r_t^{(i)}(\theta), 1-\epsilon, 1+\epsilon)\bar{A}_t^{(i)}\right)\right]. \tag{8}$$

This objective maintains a clean separation of concerns: $m_t^{(i)}$ determines which positions enter the update, $w_t^{(i)}$ determines credit magnitude at those positions, and $A^{(i)}$ is the sole sequence-level signal determining update direction. Here ϵ is the standard PPO likelihood-ratio clipping bound, distinct from ϵ_w (the reweighting clip bound introduced above). Proposition B.3 (Appendix B) formalizes this credit concentration: under the empirical selector precision $q = 0.79$ (Section 4.5) and $\epsilon_w = 0.2$, at least 71.5% of weighted update mass is directed to fork tokens, compared to the fork-token rate of $f \approx 21\%$ under a full-token update—a concentration ratio of at least $3.4\times$. Appendix B proves the corresponding structural properties of direction preservation and credit concentration under explicit assumptions. The full training procedure is in Appendix H.

4 Experiments

Table 1: Main results on BFCL v3/v4 and Tau3 (backbone: Qwen3-4B-Thinking-2507 [27]). GRPO, DAPO, and CFT use the same structured fine reward ($C = 3$); distillation baselines use their standard KL objectives. Tau3 Overall is the average over Retail, Airline, and Telecom. $\pm$ denotes mean $\pm$ std over 3 seeds. **Bold** and underlined indicate best and second-best in Multi Turn, BFCL Overall, and Tau3 Overall. Slash-separated values report BFCL v3 / v4 scores (e.g., $84.03/84.10$ denotes v3 / v4 accuracy). BFCL v4 Overall uses a stricter multi-category aggregation that yields lower absolute values ($\approx$40% range vs. $\approx$70% for v3); comparisons should be made within each version.

Model	BFCL v3/v4 Benchmark				Tau3 Benchmark			
	Non-Live AST	Live	Multi Turn	Overall Acc	Retail	Airline	Telecom	Overall
Reference Baselines								
Base Model	$84.03_{\pm 4.50}/84.10_{\pm 6.40}$	$79.55_{\pm 4.22}/82.36_{\pm 1.84}$	$43.54_{\pm 3.39}/51.37_{\pm 0.44}$	$69.08_{\pm 1.77}/40.39_{\pm 0.08}$	$48.50_{\pm 1.11}$	$53.20_{\pm 0.66}$	$27.90_{\pm 1.48}$	$43.20_{\pm 0.65}$
Expert Model	$87.83_{\pm 1.12}/86.72_{\pm 3.13}$	$81.42_{\pm 1.65}/82.36_{\pm 1.88}$	$47.42_{\pm 1.98}/53.16_{\pm 1.16}$	$72.04_{\pm 0.10}/41.51_{\pm 0.25}$	$53.40_{\pm 0.51}$	$53.30_{\pm 0.58}$	$28.10_{\pm 1.38}$	$44.93_{\pm 0.53}$
SFT	$79.03_{\pm 8.44}/87.18_{\pm 2.22}$	$45.11_{\pm 3.21}/74.71_{\pm 1.32}$	$2.33_{\pm 2.02}/1.29_{\pm 0.94}$	$37.51_{\pm 2.97}/18.50_{\pm 1.41}$	$22.30_{\pm 1.92}$	$25.50_{\pm 1.71}$	$8.00_{\pm 1.00}$	$18.60_{\pm 0.92}$
RLVR Baselines								
GRPO	$85.74_{\pm 5.41}/87.48_{\pm 1.27}$	$80.74_{\pm 2.41}/82.04_{\pm 1.22}$	$47.17_{\pm 0.69}/52.46_{\pm 0.73}$	$71.25_{\pm 0.37}/41.25_{\pm 0.26}$	$55.60_{\pm 0.24}$	$54.20_{\pm 0.40}$	$24.50_{\pm 1.84}$	$44.77_{\pm 0.63}$
DAPO	$85.59_{\pm 5.31}/81.96_{\pm 6.04}$	$81.15_{\pm 2.19}/80.88_{\pm 0.50}$	$47.17_{\pm 0.69}/\underline{53.21}_{\pm 0.85}$	$71.46_{\pm 0.66}/41.25_{\pm 0.37}$	$54.90_{\pm 0.31}$	$55.60_{\pm 0.23}$	$27.10_{\pm 1.64}$	$\underline{45.87}_{\pm 0.56}$
Distillation Baselines								
SDPO	$84.72_{\pm 4.58}/85.96_{\pm 2.75}$	$82.51_{\pm 1.56}/80.98_{\pm 0.78}$	$46.75_{\pm 1.35}/52.96_{\pm 0.97}$	$71.64_{\pm 0.71}/\underline{41.53}_{\pm 0.27}$	$51.20_{\pm 0.92}$	$53.20_{\pm 0.68}$	$28.50_{\pm 1.30}$	$44.30_{\pm 0.58}$
OPSD	$80.94_{\pm 7.85}/84.30_{\pm 5.19}$	$82.21_{\pm 2.99}/80.98_{\pm 0.34}$	$44.83_{\pm 2.08}/52.58_{\pm 0.92}$	$69.79_{\pm 0.31}/41.23_{\pm 0.16}$	$49.20_{\pm 1.04}$	$54.90_{\pm 0.30}$	$28.10_{\pm 1.37}$	$44.07_{\pm 0.58}$
On-policy distillation	$87.83_{\pm 0.89}/86.96_{\pm 0.59}$	$80.53_{\pm 3.27}/81.15_{\pm 0.11}$	$\underline{47.79}_{\pm 2.00}/52.33_{\pm 0.51}$	$\underline{71.89}_{\pm 0.52}/41.38_{\pm 0.14}$	$52.80_{\pm 0.73}$	$52.20_{\pm 0.87}$	$27.80_{\pm 1.55}$	$44.27_{\pm 0.64}$
Our Method								
CFT (Ours)	$88.66_{\pm 0.74}/88.74_{\pm 1.25}$	$82.84_{\pm 0.72}/82.48_{\pm 0.60}$	$\mathbf{51.41}_{\pm 0.36}/\mathbf{59.00}_{\pm 0.43}$	$\mathbf{74.37}_{\pm 0.29}/\mathbf{43.41}_{\pm 0.16}$	$61.30_{\pm 0.07}$	$61.00_{\pm 0.15}$	$37.00_{\pm 1.20}$	$\mathbf{53.10}_{\pm 0.40}$

4.1 Experimental Setup

Training We construct a 20k-sample training set by sampling from xlam1 [16], xlam2 [18], and ToolACE [14], covering diverse tool-use scenarios and complexity levels. All RL methods use the same configuration: batch size 128, 8 samples per prompt, temperature 1.0, 2 epochs; GRPO KL coefficient 0.01. For CFT, $\epsilon_w = 0.2$. The teacher's privileged information is instantiated as the training-time ground-truth per-step reference output. We fix the structured tool-call correctness from Section 3.3 unless noted otherwise (Section 4.6): RL methods use sequence-level fine reward ($C = 3$); distillation baselines use their standard KL objectives. Table 5 demonstrates that fine-grained verifier feedback and selective token credit are complementary, isolating the contribution of each component. Results are mean(std) over 3 independent training seeds except Table 7 and Appendix G; for Tau3, within each seed we average pass@1 over 3 runs per task before aggregating across seeds. Full hyperparameters and benchmark evaluation details are in Appendix C.

Evaluation We evaluate on BFCL v3/v4 [17] and Tau3 [4]. BFCL measures executable function-call accuracy across function selection, argument generation, and multi-step tool interaction; Tau3 evaluates conversational agents in dual-control settings. For Tau3, maximum 200 execution steps per task; both the user simulator (Qwen3-Next-80B-A3B-Thinking) and the evaluated agent are decoded with temperature 0.6, top-p 0.95, top-k 20.

Baselines We compare three groups: reference baselines (Base Model, Expert Model, SFT); RLVR baselines (GRPO [21], DAPO [30]); and distillation baselines (OPSD [33], SDPO [7], On-policy

distillation [1]). Expert Model and On-policy distillation both use Qwen3-30B-A3B-Thinking-2507; SDPO, OPSD, and CFT use same-parameter teacher (Qwen3-4B-Thinking-2507). The larger-capacity Qwen3-8B teacher appears only in Section 4.4 as a capacity control, with and without answer access. SFT uses single-turn training data; its multi-turn collapse reflects format distribution mismatch rather than limited model capacity, and serves as a reference lower bound illustrating the importance of multi-turn format alignment.

4.2 Main Results

CFT improves BFCL multi-turn accuracy by +4.2/+6.5 pp and Tau3 average by +8.3 pp over GRPO, with overall BFCL gains of +3.1/+2.2 pp in a dense leaderboard where adjacent models typically differ by only 0.1–0.4 pp. Since GRPO, DAPO, and CFT share the same structured fine reward, this gap is attributable to the selective credit mechanism; gains concentrate in sequential tool decision scenarios, consistent with the method's motivation.

CFT further surpasses the Expert Model (BFCL overall: 72.04/41.51) and On-policy distillation—which employs this same Expert Model as a larger-capacity teacher—by +3.6/+6.7 pp on multi-turn and $\sim$+8.8 pp on Tau3, confirming that answer-conditioned fork localization, not teacher capacity, drives the gains.

4.3 Decomposing Where and How Much to Update

This section answers a core question: do gains come from where to update, how much to update, or simply from stronger KL guidance?

Table 2: Core design ablations covering Selection vs. Weighting, KL Direction, and Support Set. `KL weighting only` applies full-token KL weighting without the IG_t mask. Tau3 Overall is the average of Retail, Airline, and Telecom domain scores.

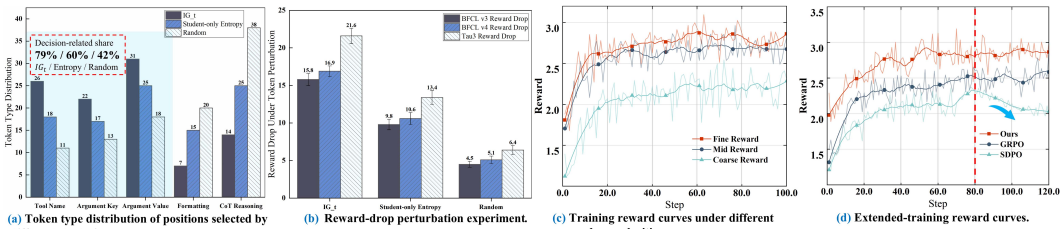
Group	Setting	BFCL v3 Overall	BFCL v4 Overall	Tau3 Overall
Selection vs Weighting	GRPO	$71.25_{\pm 0.37}$	$41.25_{\pm 0.26}$	$44.77_{\pm 0.63}$
	GRPO + IG_t mask only	$73.81_{\pm 0.20}$	$42.63_{\pm 0.22}$	$47.52_{\pm 0.56}$
	GRPO + KL weighting only	$73.56_{\pm 0.25}$	$42.46_{\pm 0.23}$	$47.88_{\pm 0.59}$
	GRPO + IG_t mask + KL weighting (Ours)	$\mathbf{74.37_{\pm 0.29}}$	$\mathbf{43.41_{\pm 0.16}}$	$\mathbf{53.10_{\pm 0.40}}$
KL Direction	Symmetric KL	$73.92_{\pm 0.24}$	$42.89_{\pm 0.20}$	$50.96_{\pm 0.49}$
	Asymmetric KL (Ours)	$\mathbf{74.37_{\pm 0.29}}$	$\mathbf{43.41_{\pm 0.16}}$	$\mathbf{53.10_{\pm 0.40}}$
Support Set	Teacher-only support	$74.14_{\pm 0.22}$	$43.15_{\pm 0.19}$	$52.34_{\pm 0.44}$
	Union-only support (Ours)	$\mathbf{74.37_{\pm 0.29}}$	$\mathbf{43.41_{\pm 0.16}}$	$\mathbf{53.10_{\pm 0.40}}$

Table 2 supports three conclusions. First, IG_t selection and KL weighting each contribute individually, and their combination is best, confirming that where to update and how much to update are complementary. Second, asymmetric KL consistently outperforms symmetric KL, validating the direction-sensitive fork error design. Third, union support consistently outperforms teacher-only support, indicating that effective credit reweighting requires both teacher-preferred branches and student high-probability branches.

Table 3: Token selector comparison under matched training settings.

Setting	BFCL v3 Overall	BFCL v4 Overall	Tau3 Overall
Random mask	$72.81_{\pm 0.30}$	$42.07_{\pm 0.21}$	$43.11_{\pm 0.63}$
Full token	$73.56_{\pm 0.25}$	$42.46_{\pm 0.23}$	$47.88_{\pm 0.59}$
Student-only entropy	$73.41_{\pm 0.20}$	$42.13_{\pm 0.18}$	$46.08_{\pm 0.57}$
Privileged-information-conditioned IG_t mask (Ours)	$\mathbf{74.37_{\pm 0.29}}$	$\mathbf{43.41_{\pm 0.16}}$	$\mathbf{53.10_{\pm 0.40}}$

Table 3 shows IG_t outperforms all alternatives with the largest gap on Tau3, confirming gains come from targeted updates to correctable fork tokens. Having established that both components contribute, we next ask whether gains arise from answer conditioning or teacher capacity.



(a) Token type distribution of positions selected by different strategies. (b) Reward-drop perturbation experiment. (c) Training reward curves under different reward granularities (d) Extended-training reward curves.

Figure 2: Performance and mechanism analysis. (a,b) Mechanism validation: IG_t-selected positions concentrate on structured decision tokens and cause the largest reward drops under perturbation. (c) Reward granularity: finer-grained rewards improve faster and reach higher plateaus, consistently following fine $>$ mid $>$ coarse. (d) Training stability: CFT maintains the highest reward over extended training, while SDPO begins to decline around step 80.

4.4 Why Answer Conditioning Matters

This section separates the effects of teacher capacity and answer access with a complete 2×2 comparison.

Table 4: Complete 2×2 answer-conditioning controls over teacher capacity and answer access, with a same-capacity noisy-answer control.

Setting	BFCL v3 Overall	BFCL v4 Overall	Tau3 Overall
same-capacity + no-answer teacher	$73.41_{\pm 0.20}$	$42.13_{\pm 0.18}$	$46.08_{\pm 0.57}$
same-capacity + noisy-answer teacher	$72.78_{\pm 0.23}$	$41.74_{\pm 0.19}$	$42.87_{\pm 0.66}$
same-capacity + answer-conditioned teacher	$\mathbf{74.37_{\pm 0.29}}$	$\mathbf{43.41_{\pm 0.16}}$	$\mathbf{53.10_{\pm 0.40}}$
larger-capacity + no-answer teacher	$73.53_{\pm 0.21}$	$42.24_{\pm 0.18}$	$46.47_{\pm 0.59}$
larger-capacity + answer-conditioned teacher	$74.29_{\pm 0.31}$	$43.28_{\pm 0.20}$	$52.44_{\pm 0.51}$

The "same-capacity + no-answer" condition is equivalent to "student-only entropy" (Table 3): without privileged information, $H_T = H_S$, so $IG_t = 0$; therefore, selection adopts entropy-based masking, as confirmed by the identical reported numbers. Table 4 shows that increasing capacity from 4B to 8B changes performance by only $+0.12/+0.11/+0.39$ pp without answer access and $-0.08/-0.13/-0.66$ pp with answer access. In contrast, answer conditioning improves the 4B setting by $+0.96/+1.28/+7.02$ pp and the 8B setting by $+0.76/+1.04/+5.97$ pp. The 8B answer-conditioned teacher does not improve over the shared-parameter 4B design in this measured setting. The noisy-answer condition replaces the ground-truth answer with a randomly sampled answer from a different training example, preserving structural format while corrupting semantic content. It performs worse than no-answer conditioning because the mismatched signal changes the entropy landscape and moves credit to different positions.

A same-parameter teacher is structurally motivated and sufficient in the evaluated setting: sharing the representation space with the student ensures that IG_t measures entropy reduction caused by the answer rather than capacity-induced distribution shifts between heterogeneous models. The complete control shows that answer access is the main factor and that increased capacity alone does not reproduce CFT's gains.

4.5 Mechanism Validation

Section 3.4 argued structurally that high-IG_t positions correspond to correctable decision forks; this section tests that claim empirically from two angles: descriptive statistics of selected token types and causal perturbation experiments.

As shown in Figure 2a, decision-related tokens (Tool Name + Argument Key + Argument Value) comprise 79% of IG_t-selected positions, versus 60% for student-only entropy and 42% for random, demonstrating that IG_t concentrates on the structured decision tokens that genuinely determine subsequent execution paths. The remaining 21% are predominantly high-entropy natural-language

positions that coincidentally reach high IG_t values; these do not introduce systematic noise into the credit signal. As a further check, Table 19 removes the 13.4% of selected positions that overlap with answer surface tokens—the subset most likely to reflect surface copying—and observes only a 0.54 pp drop on Tau3, confirming that surface-token overlap does not drive CFT's gains.

Table 5: Joint ablation of reward granularity and credit scope. Each cell reports BFCL v3 / BFCL v4 / Tau3. Fine-grained verifier feedback and selective token credit are complementary. *Full top-k* applies KL-based credit reweighting to all tokens without the IG_t selection mask, corresponding to *KL weighting only* in Table 2 and *Full token* in Table 3. *Note*: The Sequence-level credit scope applies answer-conditioned KL reweighting uniformly over all tokens; it is not directly comparable to the GRPO baseline in Table 1, which applies no per-token KL reweighting.

Reward Granularity	Sequence-level	Full top-k	Selected token (Ours)
coarse reward	$73.00_{\pm 0.23}/41.88_{\pm 0.20}/44.61_{\pm 0.64}$	$73.24_{\pm 0.21}/42.11_{\pm 0.18}/46.31_{\pm 0.58}$	$73.49_{\pm 0.20}/42.33_{\pm 0.17}/48.06_{\pm 0.54}$
mid reward	$73.29_{\pm 0.22}/42.18_{\pm 0.18}/45.93_{\pm 0.59}$	$73.48_{\pm 0.20}/42.39_{\pm 0.18}/46.85_{\pm 0.58}$	$73.88_{\pm 0.18}/42.78_{\pm 0.17}/50.47_{\pm 0.46}$
fine reward	$73.39_{\pm 0.21}/42.28_{\pm 0.19}/46.84_{\pm 0.57}$	$73.56_{\pm 0.19}/42.46_{\pm 0.17}/47.88_{\pm 0.42}$	$\mathbf{74.37}_{\pm 0.29}/\mathbf{43.41}_{\pm 0.16}/\mathbf{53.10}_{\pm 0.40}$

Table 6: Scale robustness validation on an independent Qwen3 model family (BFCL v4 overall). This series uses a different Qwen3 backbone (4B/8B/14B) from Table 1 and is not directly comparable.

Model Size	GRPO	SDPO	Ours
4B	$32.41_{\pm 0.20}$	$34.25_{\pm 0.24}$	$35.68_{\pm 0.16}$
8B	$38.67_{\pm 0.18}$	$40.79_{\pm 0.21}$	$42.24_{\pm 0.18}$
14B	$40.22_{\pm 0.17}$	$42.57_{\pm 0.19}$	$45.34_{\pm 0.13}$

Table 7: Wall-clock time to reach a fixed BFCL target, defined as the earliest checkpoint achieving $\geq$70% on BFCL v3 and $\geq$40% on BFCL v4, measured on 4×A100 GPUs. Tau3 evaluation is excluded due to higher per-evaluation cost. Wall-clock time per full training update is 3.24 min for GRPO and 3.62 min for SDPO/Ours. Updates-to-target use unrounded measurements; the displayed per-update times are rounded. Ours reaches the target in 74 updates, fewer than GRPO (117) and SDPO (82), and has the best measured time-to-target.

Model	Time-to-Target (Hours)	Wall-Clock / Update	Updates-to-Target
GRPO	6.31	3.24 min	117
SDPO	4.97	3.62 min	82
Ours	**4.51**	**3.62 min**	**74**

This concentration cannot be explained by surface answer copying: the overlap rate between IG_t-selected positions and answer surface tokens is only 13.4% (Appendix E Table 18), far below what a surface-copy explanation would predict.

As shown in Figure 2b, at each selected position, the realized token is replaced by the highest-probability alternative in the local top-k, and the rollout continues from the perturbed prefix. Perturbing IG_t-selected positions causes the largest reward drops on BFCL v3/v4/Tau3: 15.8 / 16.9 / 21.6, providing causal evidence that these positions are genuine execution-path forks. Supplementary analyses in Appendix E.

4.6 Additional Analyses

Auxiliary Cross-Scale Evidence Table 6 reports results on an independent Qwen3 series, distinct from the Table 1 backbone, and provides within-family cross-scale evidence beyond a single backbone instance. CFT leads at all three scales on this independent series, and the margin over SDPO grows from +1.43 to +2.77 pp, suggesting larger backbones may benefit more from answer-conditioned fork-token selection.

Reward Granularity and Training Dynamics As shown in Figure 2c, finer-grained rewards produce faster reward improvement over most of training and reach a higher plateau, consistently following the ordering fine $>$ mid $>$ coarse. As shown in Table 5, when credit scope is fixed, performance follows coarse $<$ mid $<$ fine; when reward granularity is fixed, performance follows sequence-level $<$

full top-$k <$ selected token. This indicates that, for tool-call settings, finer-grained verifier feedback and more precise selected-token credit are complementary.

Training Stability and Compute Overhead As shown in Figure 2d, CFT maintains the highest reward throughout and remains stable; GRPO stays below CFT; SDPO rises faster initially but begins declining around step 80—consistent with analyses by Kim et al. [9] and Fu et al. [5] on same-parameter self-distillation: as the student distribution drifts from the initial policy, teacher guidance on student-generated prefixes becomes unreliable, and the progressive suppression of uncertainty expression incurs a generalization cost over long training. As shown in Table 7, the answer-conditioned branch overhead is offset by higher optimization efficiency.

Lightweight OOD Sanity Check General capability and OOD results are reported in Appendix G Table 24. IFEval and MMLU-Pro slightly improve; AIME25 changes by only -0.4 pp. General capabilities remain stable with no systematic degradation.

Additional controls are reported in Appendix D: structural-mask and step-level credit baselines, success-regime analysis, cross-family and non-tool transfer results, pseudo-answer and noisy-answer controls, and a decomposition of compute and memory overhead.

5 Conclusion

This work addresses the credit assignment mismatch in tool-integrated RLVR between sequence-level verifier rewards and sparse tool-call decision tokens. We propose Correctable Fork Tokens: a same-parameter, answer-conditioned branch localizes correctable decision forks via IG_t; asymmetric KL rescales token-level credit magnitude at those positions; and the verifier-anchored update direction is preserved throughout. We additionally present a three-level structured verifier for tool-call settings that is independently applicable and complementary to CFT. Experiments demonstrate substantial gains on multi-turn and Tau3 settings with stable overall BFCL improvements. Ablations, answer-conditioning controls, and mechanism analysis confirm that gains stem from precise localization of decision-critical tool-call positions—not from teacher capacity—and that finer-grained verifier feedback and selective token credit are complementary. A natural next step is extending CFT to open-ended agentic settings where ground-truth answers are unavailable, using self-consistency signals or step-level execution feedback as privileged information.

6 Limitations and Broader Impacts

CFT requires ground-truth answers at training time; extending to open-ended agentic settings via self-consistency or ensemble signals is a natural direction. Applying CFT to broader model families and exploring larger same-family teachers for improved IG_t precision are promising future directions. Current IG_t concentrates on uncertain positions; capturing confident but systematically wrong decisions is an open extension.

Improving credit assignment for tool-integrated RLVR benefits LLM-based agent reliability in real-world API, enterprise automation, and conversational service settings. As with any method increasing agent capability, responsible deployment should pair this training technique with appropriate permission controls and safety checks.

Acknowledgments and Disclosure of Funding

This work was completed independently by the author and received no third-party funding or resources. The author is employed by Li Auto in an agent-related role; Li Auto had no role in this work. The author reports no other competing financial interests.

References

[1] Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and

Y. Sun, editors, *International Conference on Learning Representations*, volume 2024, pages 21246–21263, 2024.

[2] Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12248–12267, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.662.

[3] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. arXiv preprint arXiv:1707.01495, 2018.

[4] Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-Bench: Evaluating conversational agents in a dual-control environment. arXiv preprint arXiv:2506.07982, 2025.

[5] Yuqian Fu, Haohuan Huang, Kaiwen Jiang, Yuanheng Zhu, and Dongbin Zhao. Revisiting on-policy distillation: Empirical failure modes and simple fixes. arXiv preprint arXiv:2603.25562, 2026.

[6] Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (ReST) for language modeling. arXiv preprint arXiv:2308.08998, 2023.

[7] Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement learning via self-distillation. arXiv preprint arXiv:2601.20802, 2026.

[8] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. arXiv preprint arXiv:2503.09516, 2025.

[9] Jeonghye Kim, Xufang Luo, Minbeom Kim, Sangmook Lee, Dohyung Kim, Jiwon Jeon, Dongsheng Li, and Yuqing Yang. Why does self-distillation (sometimes) degrade the reasoning capability of LLMs? arXiv preprint arXiv:2603.24472, 2026.

[10] Xuefeng Li, Haoyang Zou, and Pengfei Liu. ToRL: Scaling tool-integrated RL. arXiv preprint arXiv:2503.23383, 2025.

[11] 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. arXiv preprint arXiv:2305.20050, 2023.

[12] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[13] Zicheng Lin, Tian Liang, Jiahao Xu, Xing Wang, Ruilin Luo, Chufan Shi, Siheng Li, Yujiu Yang, and Zhaopeng Tu. Critical tokens matter: Token-level contrastive estimation enhances LLM's reasoning capability. *arXiv preprint arXiv:2411.19943*, 2024.

[14] Weiwen Liu, Xu Huang, Xingshan Zeng, Xinlong Hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, et al. ToolACE: Winning the points of LLM function calling. In *International Conference on Learning Representations*, volume 2025, pages 41359–41381, 2025.

[15] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as agents. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun, editors, *International Conference on Learning Representations*, volume 2024, pages 52989–53046, 2024.

[16] Zuxin Liu, Thai Hoang, Jianguo Zhang, Ming Zhu, Tian Lan, Shirley Kokane, Juntao Tan, Weiran Yao, Zhiwei Liu, Yihao Feng, Rithesh Murthy, Liangwei Yang, Silvio Savarese, Juan Carlos Niebles, Huan Wang, Shelby Heinecke, and Caiming Xiong. APIGen: Automated pipeline for generating verifiable and diverse function-calling datasets. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 54463–54482. Curran Associates, Inc., 2024. doi: 10.52202/079017-1725.

[17] Shishir G. Patil, Huanzhi Mao, Charlie Cheng-Jie Ji, Fanjia Yan, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The Berkeley Function Calling Leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*, 2025.

[18] Akshara Prabhakar, Zuxin Liu, Weiran Yao, Jianguo Zhang, Ming Zhu, Shiyu Wang, Zhiwei Liu, Tulika Manoj Awalgaonkar, Haolin Chen, Thai Hoang, Juan Carlos Niebles, Shelby Heinecke, Huan Wang, Silvio Savarese, and Caiming Xiong. APIGen-MT: Agentic pipeline for multi-turn data generation via simulated agent-human interplay. *arXiv preprint arXiv:2504.03601*, 2025.

[19] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. arXiv preprint arXiv:2307.16789, 2023.

[20] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. arXiv preprint arXiv:2302.04761, 2023.

[21] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300, 2024.

[22] Charles Burton Snell, Dan Klein, and Ruiqi Zhong. Learning by distilling context. *arXiv preprint arXiv:2209.15189*, 2022.

[23] Richard S. Sutton. Learning to predict by the methods of temporal differences. *Mach. Learn.*, 3 (1):9–44, August 1988.

[24] Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural Netw.*, 22(5–6):544–557, July 2009.

[25] Jean Vassoyan, Nathanaël Beau, and Roman Plaud. Ignore the KL penalty! Boosting exploration on critical tokens to enhance RL fine-tuning. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 6123–6133, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.340.

[26] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256, May 1992.

[27] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, et al. Qwen3 technical report. arXiv preprint arXiv:2505.09388, 2025.

[28] Feng Yao, Liyuan Liu, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. Your efficient RL framework secretly brings you off-policy RL training. Blog post, `https://fengyao.notion.site/off-policy-rl`, August 2025.

[29] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

[30] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. arXiv preprint arXiv:2503.14476, 2025.

[31] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STaR: Bootstrapping reasoning with reasoning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 15476–15488. Curran Associates, Inc., 2022.

[32] Shaokun Zhang, Yi Dong, Jieyu Zhang, Jan Kautz, Bryan Catanzaro, Andrew Tao, Qingyun Wu, Zhiding Yu, and Guilin Liu. Nemotron-Research-Tool-N1: Exploring tool-using language models with reinforced reasoning. arXiv preprint arXiv:2505.00024, 2025.

[33] Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. arXiv preprint arXiv:2601.18734, 2026.

A Exact KL and Numerical Details

A.1 KL Formulas and Degenerate Case

The compact KL notation used in the main text has the following precise implementation:

$$D_{KL}(\tilde{P}_S^t \| \tilde{P}_T^t) = \sum_{v \in \mathcal{V}_t^K} \tilde{P}_S^t(v) \log \frac{\tilde{P}_S^t(v)}{\tilde{P}_T^t(v)} + \tilde{P}_S^t(\text{tail}) \log \frac{\tilde{P}_S^t(\text{tail})}{\tilde{P}_T^t(\text{tail})}, \tag{9}$$

$$D_{KL}(\tilde{P}_T^t \| \tilde{P}_S^t) = \sum_{v \in \mathcal{V}_t^K} \tilde{P}_T^t(v) \log \frac{\tilde{P}_T^t(v)}{\tilde{P}_S^t(v)} + \tilde{P}_T^t(\text{tail}) \log \frac{\tilde{P}_T^t(\text{tail})}{\tilde{P}_S^t(\text{tail})}. \tag{10}$$

In both equations, all distribution terms ($\tilde{P}_S^t$ and $\tilde{P}_T^t$) are computed with stop-gradient; the resulting $s_t^{(i)}$ and $\bar{A}_t^{(i)}$ are scalar multipliers in the policy gradient objective and introduce no additional gradient paths.

If a trajectory satisfies

$$\sum_{j \in S^{(i)}} s_j^{(i)} = 0, \tag{11}$$

then we set

$$w_t^{(i)} = 1 \tag{12}$$

for all positions in that trajectory. In this case the objective reduces to the unweighted sequence-level advantage form.

A.2 Reward Granularity Variants

The three reward granularity variants in Table 5 share the matched-call framework from Section 3.3: they use the same greedy one-to-one matching M and the same union-style denominator $m+n-|M|$, differing only in per-pair scoring.

fine reward uses the default definition from Section 3.3:

$$s_{ij}^{\text{fine}} = 1 + r_{\text{argname}}(p_i, g_j) + r_{\text{argvalue}}(p_i, g_j) \in [1, 3], \tag{13}$$

$$R_{\text{Tools}}^{\text{fine}} = \begin{cases} 3, & m = n = 0, \\ \dfrac{\sum_{(i,j) \in M} s_{ij}^{\text{fine}}}{m + n - |M|}, & \text{otherwise}, \end{cases} \qquad R_{\text{correct}}^{\text{fine}} = C \cdot \left(2 \cdot \frac{R_{\text{Tools}}^{\text{fine}}}{3} - 1 \right). \tag{14}$$

mid reward removes the argument-value layer and retains only argument-name supervision within each matched pair:

$$s_{ij}^{\text{mid}} = 1 + r_{\text{argname}}(p_i, g_j) \in [1, 2], \tag{15}$$

$$R_{\text{Tools}}^{\text{mid}} = \begin{cases} 2, & m = n = 0, \\ \dfrac{\sum_{(i,j) \in M} s_{ij}^{\text{mid}}}{m + n - |M|}, & \text{otherwise}, \end{cases} \qquad R_{\text{correct}}^{\text{mid}} = C \cdot \left(2 \cdot \frac{R_{\text{Tools}}^{\text{mid}}}{2} - 1 \right). \tag{16}$$

coarse reward merges argument name and argument value into an argument-level exact match:

$$r_{\text{args}}^{\text{coarse}}(p_i, g_j) = \mathbf{1}[K(p_i) = K(g_j) \ \wedge \ \forall k \in K(g_j),\ s_{\text{val}}(v(p_i, k), v(g_j, k)) = 1], \tag{17}$$

$$s_{ij}^{\text{coarse}} = 1 + r_{\text{args}}^{\text{coarse}}(p_i, g_j) \in [1, 2], \tag{18}$$

$$R_{\text{Tools}}^{\text{coarse}} = \begin{cases} 2, & m = n = 0, \\ \dfrac{\sum_{(i,j)\in M} s_{ij}^{\text{coarse}}}{m + n - |M|}, & \text{otherwise}, \end{cases} \qquad R_{\text{correct}}^{\text{coarse}} = C \cdot \left(2 \cdot \frac{R_{\text{Tools}}^{\text{coarse}}}{2} - 1 \right). \tag{19}$$

B Theoretical Properties of CFT

This appendix records structural properties of the CFT surrogate objective. These results are intentionally conservative: they do not claim convergence, sample-efficiency, generalization, or guaranteed reward improvement. They only formalize what the objective guarantees under the stated assumptions.

Assumption B.1 (Local surrogate analysis)**.** Consider a fixed rollout batch and a fixed sampled trajectory of length T. The notation below uses one trajectory for readability; the same counting arguments apply to the full batch by replacing $\{1, \ldots, T\}$ with the batch-level position index set. The teacher distributions are stop-gradient quantities. The selected set $S = \{t : m_t = 1\}$ is nonempty. Softmax probabilities are strictly positive on the vocabulary, so all KL terms are finite. The CFT reweighting clip satisfies $0 \leq \epsilon_w < 1$. The statements below concern the advantage coefficients and the local first-order behavior of the clipped surrogate; they do not assert any global optimization guarantee.

Proposition B.2 (Direction preservation)**.** *Under Assumption B.1, CFT cannot reverse the verifier-specified reinforcement or suppression direction. For every selected token, define*

$$\alpha_t = \text{clip}(w_t, 1 - \epsilon_w, 1 + \epsilon_w), \qquad \bar{A}_t = \alpha_t A. \tag{20}$$

Then $\text{sign}(\bar{A}_t) = \text{sign}(A)$.

Proof. Because $0 \leq \epsilon_w < 1$, the clipping interval satisfies $1 - \epsilon_w > 0$. Therefore $\alpha_t \in [1 - \epsilon_w, 1 + \epsilon_w]$ is strictly positive for every selected token. Hence

$$\text{sign}(\bar{A}_t) = \text{sign}(\alpha_t A) = \text{sign}(A). \tag{21}$$

In the unclipped branch of the surrogate, the local first-order coefficient multiplying the sampled-token likelihood ratio has the same sign as A. In the clipped branch, PPO-style clipping may make the active branch locally flat with respect to the likelihood ratio, but it does not introduce a coefficient with the opposite sign. Thus the answer-conditioned teacher can change only the update magnitude through m_t and α_t; it cannot turn a verifier-positive trajectory into a teacher-induced penalty, or a verifier-negative trajectory into a teacher-induced reward. □

Proposition B.3 (Credit concentration under high-precision selection)**.** *Let $F \subseteq \{1, \ldots, T\}$ be the set of fork tokens in a trajectory and let $S \subseteq \{1, \ldots, T\}$ be the CFT selected set. Define the selector precision on fork tokens as*

$$q = \frac{|S \cap F|}{|S|}, \qquad f = \frac{|F|}{T}. \tag{22}$$

For unweighted position-level credit, CFT assigns fraction q of its selected-token update positions to fork tokens, while a full-token sequence-level update assigns fraction f. Therefore, if $q > f$, masked selection concentrates position-level credit on fork tokens relative to a full-token update. With clipped CFT weights $\alpha_t \in [1 - \epsilon_w, 1 + \epsilon_w]$, the weighted fork-token mass satisfies

$$\frac{\sum_{t \in S \cap F} \alpha_t}{\sum_{t \in S} \alpha_t} \geq \frac{q(1 - \epsilon_w)}{q(1 - \epsilon_w) + (1 - q)(1 + \epsilon_w)}. \tag{23}$$

Proof. The unweighted statement follows directly from counting. CFT updates positions only in S, so the fraction of selected update positions that are fork tokens is $|S \cap F|/|S| = q$. A full-token sequence-level update spreads position-level credit over all T positions, so the corresponding fraction is $|F|/T = f$.

For the weighted statement, write

$$W_F = \frac{\sum_{t \in S \cap F} \alpha_t}{\sum_{t \in S \cap F} \alpha_t + \sum_{t \in S \setminus F} \alpha_t}. \tag{24}$$

Since $\alpha_t \in [1-\epsilon_w, 1+\epsilon_w]$ for all selected tokens uniformly, we apply the global lower bound $1-\epsilon_w$ to fork-token terms (appearing in the numerator) and the global upper bound $1+\epsilon_w$ to non-fork terms (appearing only in the denominator); this yields the tightest lower bound on W_F:

$$\sum_{t \in S \cap F} \alpha_t \geq |S \cap F|(1-\epsilon_w), \qquad \sum_{t \in S \setminus F} \alpha_t \leq (|S| - |S \cap F|)(1+\epsilon_w). \tag{25}$$

The ratio $a/(a+b)$ is increasing in a and decreasing in b for nonnegative a, b with $a + b > 0$. Substituting the lower bound for the fork-token numerator and the upper bound for the non-fork selected mass gives

$$W_F \geq \frac{|S \cap F|(1-\epsilon_w)}{|S \cap F|(1-\epsilon_w) + (|S| - |S \cap F|)(1+\epsilon_w)} = \frac{q(1-\epsilon_w)}{q(1-\epsilon_w) + (1-q)(1+\epsilon_w)}. \tag{26}$$

□

Remark B.4 (Empirical selector precision is not part of the proof). Proposition B.3 is conditional on selector precision q. The empirical result in Section 4.5 that 79% of IG_t-selected positions are decision-related tokens is not used as a theoretical proof. It only instantiates the condition in observed training rollouts: the IG_t selector operates in a high-precision regime relative to student-only entropy and random masking, so the proposition's credit-concentration premise is empirically plausible for this setting.

Lemma B.5 (IG_t as conditional mutual information in expectation)**.** *Let $C_t = (x, y_{<t})$ denote the prefix context, let Z denote the privileged information, and let Y_t be the next token. Suppose the student and teacher distributions are calibrated conditionals,*

$$P_S^t(\cdot) = P(Y_t = \cdot \mid C_t), \qquad P_T^t(\cdot) = P(Y_t = \cdot \mid C_t, Z). \tag{27}$$

Then the population expectation of the information gap satisfies

$$\mathbb{E}[IG_t] = H(Y_t \mid C_t) - H(Y_t \mid C_t, Z) = I(Y_t; Z \mid C_t) \geq 0. \tag{28}$$

Proof. By definition, for a realized context and privileged signal,

$$IG_t = H(P_S^t) - H(P_T^t). \tag{29}$$

Under the calibration assumption, taking expectation over the joint distribution of (C_t, Z) gives

$$\mathbb{E}[H(P_S^t)] = \mathbb{E}_{c_t}\big[H(Y_t \mid C_t = c_t)\big] = H(Y_t \mid C_t), \tag{30}$$

and

$$\mathbb{E}[H(P_T^t)] = \mathbb{E}_{C_t, Z}\big[H(Y_t \mid C_t, Z)\big] = H(Y_t \mid C_t, Z). \tag{31}$$

Therefore

$$\mathbb{E}[IG_t] = H(Y_t \mid C_t) - H(Y_t \mid C_t, Z) = I(Y_t; Z \mid C_t). \tag{32}$$

Conditional mutual information is nonnegative. This is a population interpretation only; a realized sample-level IG_t need not be nonnegative. □

Lemma B.6 (Top-k+tail KL is conservative)**.** *Let P and Q be strictly positive distributions over a finite vocabulary $\mathcal{V}$. Let $\mathcal{P}$ be the partition that keeps each token in a support set $U \subseteq \mathcal{V}$ as a singleton and merges $\mathcal{V} \setminus U$ into one tail bucket. Define the coarse-grained distributions $\tilde{P}(B) = \sum_{v \in B} P(v)$ and $\tilde{Q}(B) = \sum_{v \in B} Q(v)$ for each bucket $B \in \mathcal{P}$. Then*

$$D_{KL}(\tilde{P} \| \tilde{Q}) \leq D_{KL}(P \| Q). \tag{33}$$

The same statement holds for the reverse direction, $D_{KL}(\tilde{Q} \| \tilde{P}) \leq D_{KL}(Q \| P)$.

Proof. For any bucket $B \in \mathcal{P}$, the log-sum inequality gives

$$\sum_{v \in B} P(v) \log \frac{P(v)}{Q(v)} \geq \left(\sum_{v \in B} P(v) \right) \log \frac{\sum_{v \in B} P(v)}{\sum_{v \in B} Q(v)} = \tilde{P}(B) \log \frac{\tilde{P}(B)}{\tilde{Q}(B)}. \tag{34}$$

Summing this inequality over all buckets in $\mathcal{P}$ yields

$$D_{KL}(P\|Q) = \sum_{B \in \mathcal{P}} \sum_{v \in B} P(v) \log \frac{P(v)}{Q(v)} \geq \sum_{B \in \mathcal{P}} \tilde{P}(B) \log \frac{\tilde{P}(B)}{\tilde{Q}(B)} = D_{KL}(\tilde{P}\|\tilde{Q}). \tag{35}$$

Swapping P and Q proves the reverse-direction statement. Thus the union top-k+tail approximation used by CFT is a coarse-grained lower bound on full-vocabulary KL; it cannot create a KL value larger than the full-vocabulary divergence for the same two distributions. □

C Experiment Details

C.1 Training Configuration

Table 8 summarizes the complete training hyperparameters shared across all RL methods. Two rollout batches (256 samples each, 512 total) are collected before each gradient update; the optimization mini-batch size is 128. The IG_t mask threshold $\tau_\rho^{\mathcal{B}}$ is computed globally over this mini-batch after excluding padding tokens. All tool calls are represented in JSON format, providing a general and structured interface compatible with both the rollout engine and the verifier.

Table 8: Training hyperparameters shared across all RL methods unless otherwise noted.

Hyperparameter	**Value**
Rollout batch size	256
Train batch size	128
Samples per prompt (G)	8
Temperature	1.0
Max prompt length	4096 tokens
Max response length	2048 tokens
Learning rate	5e-7
KL coefficient	0.01
KL estimator	k1
CFT reweighting clip (ϵ_w)	0.2
Top-k size (K)	20
Correctness weight (C)	3
Training epochs	2

To mitigate trajectory mismatch between the rollout engine (vLLM) and the training framework (FSDP)—which can cause on-policy data collection to drift toward off-policy—we apply importance sampling correction during training [28].

C.2 Benchmark Evaluation

BFCL v3/v4. The Berkeley Function Calling Leaderboard [17] is a comprehensive tool-use benchmark covering single-turn and multi-turn function calling, multi-step tool execution, irrelevant-tool rejection, and multi-tool selection. We align BFCL v4 metrics with the v3 format and use the default evaluation protocol; all tool invocations are represented in JSON.

Tau3. The Tau3 benchmark [4] evaluates conversational agents in a dual-control environment across three domains (Retail, Airline, Telecom), where both the user simulator and the agent policy are language models. Each task allows up to 200 execution steps. We use Qwen3-Next-80B-A3B-Thinking as the user simulator with decoding settings temperature 0.6, top-p 0.95, and top-k 20. For each seed, we report the average pass@1 over 3 independent task runs per domain before aggregating across seeds.

D Additional Controls and Analyses

This appendix reports additional controls on structural and temporal credit, success regimes, transfer, privileged-information sources, and compute overhead. Unless otherwise noted, results are mean(std) over three independent training seeds and use the main-text protocol; Tables 16 and 17 are single profiling runs. Tables 9–10 cover structural and temporal credit baselines; Tables 11–15 cover success regimes and transfer controls; Tables 16 and 17 decompose compute and memory overhead.

D.1 Structural Masks and Temporal Credit

Table 9: Structural-mask baselines and matched CFT weighting. S0 updates all tokens inside `<tool_call>` spans; S1 updates tool-name, argument-key, and argument-value tokens.

Setting	Size (%)	v3 Overall	v3 Multi-turn	v4 Overall	v4 Multi-turn	Tau3
GRPO	100.0	$71.25_{\pm 0.37}$	$47.17_{\pm 0.69}$	$41.25_{\pm 0.26}$	$52.46_{\pm 0.73}$	$44.77_{\pm 0.63}$
S0: span mask	33.4	$72.94_{\pm 0.31}$	$48.94_{\pm 0.62}$	$41.98_{\pm 0.24}$	$53.99_{\pm 0.68}$	$45.62_{\pm 0.61}$
S0 + CFT weighting	33.4	$73.32_{\pm 0.28}$	$49.41_{\pm 0.55}$	$42.46_{\pm 0.22}$	$55.06_{\pm 0.62}$	$48.23_{\pm 0.53}$
S1: name/key/value mask	20.8	$73.62_{\pm 0.24}$	$49.78_{\pm 0.52}$	$42.38_{\pm 0.21}$	$54.95_{\pm 0.60}$	$47.13_{\pm 0.56}$
S2: S1 + CFT weighting	20.8	$74.08_{\pm 0.23}$	$50.28_{\pm 0.44}$	$42.98_{\pm 0.19}$	$56.44_{\pm 0.52}$	$50.71_{\pm 0.48}$
IG_t mask only	20.0	$73.81_{\pm 0.20}$	$50.50_{\pm 0.48}$	$42.63_{\pm 0.22}$	$56.53_{\pm 0.55}$	$47.52_{\pm 0.56}$
CFT (Ours)	20.0	$\mathbf{74.37}_{\pm 0.29}$	$\mathbf{51.41}_{\pm 0.36}$	$\mathbf{43.41}_{\pm 0.16}$	$\mathbf{59.00}_{\pm 0.43}$	$\mathbf{53.10}_{\pm 0.40}$

At matched budget and identical weighting, CFT exceeds S2 by 2.39 pp on Tau3 and 1.13/2.56 pp on BFCL v3/v4 multi-turn. Twenty-four percent of S1 positions fall outside the current IG_t top-20% mask; the share of structural positions clearing the IG_t threshold falls from 84.1% at step 20 to 67.9% at step 200. On Tau3, the gain from CFT weighting grows with mask quality: +2.61 / +3.58 / +5.58 pp for the S0, S1, and IG_t masks.

Table 10: Temporal credit baselines. T1 applies a leave-one-out step-level advantage only to tokens generated in that step; T2 combines step-level advantage with CFT token selection.

Setting	Temporal/token scope	v3 Overall	v4 Overall	Tau3
GRPO	trajectory / all tokens	$71.25_{\pm 0.37}$	$41.25_{\pm 0.26}$	$44.77_{\pm 0.63}$
T1: step-level advantage	per step / step tokens	$72.58_{\pm 0.28}$	$41.94_{\pm 0.24}$	$47.05_{\pm 0.58}$
CFT (Ours)	trajectory / IG_t top-20%	$74.37_{\pm 0.29}$	$43.41_{\pm 0.16}$	$53.10_{\pm 0.40}$
T2: step-level + CFT	per step / IG_t top-20%	$\mathbf{75.02}_{\pm 0.24}$	$\mathbf{44.06}_{\pm 0.17}$	$\mathbf{54.63}_{\pm 0.42}$

T1 recovers 27.4% of CFT's Tau3 gain over GRPO, while T2 improves over both baselines. On failed trajectories, 86.4% of CFT-selected positions occur at or after the first erroneous call, compared with 63.2% of all GRPO positions; the fraction of updates applied to pre-error positions falls from 36.8% to 13.6%.

D.2 Success Regimes and Transfer

Table 11: Performance across empirical GRPO success regimes. Bins are fixed using the reference GRPO checkpoint's $G = 8$ rollout pass rate; paired IG_t and selection-rate values are v3 multi-turn/Tau3. Mean IG_t and selection rate are measured only at decision positions; the top-20% mask itself is computed batch-globally over all non-padding response tokens.

GRPO pass-rate bin	v3 Multi-turn GRPO → CFT	Tau3 GRPO → CFT	Mean IG_t	Selection rate (%)
$[0, 10\%]$	$4.02 \rightarrow 5.37$	$2.84 \rightarrow 5.68$	$0.12/0.15$	$52.8/55.1$
$(10\%, 50\%]$	$33.71 \rightarrow 40.55$	$31.65 \rightarrow 46.55$	$0.71/0.78$	$96.4/94.8$
$(50\%, 90\%)$	$68.85 \rightarrow 75.07$	$68.31 \rightarrow 81.69$	$0.61/0.66$	$93.5/90.2$
$[90\%, 100\%]$	$95.24 \rightarrow 96.60$	$94.05 \rightarrow 95.97$	$0.08/0.10$	$41.6/37.6$
Overall	$47.17 \rightarrow 51.41$	$44.77 \rightarrow 53.10$	$0.42/0.43$	$74.4/70.3$

The gains concentrate in intermediate-success bins, where correct and incorrect branches are both represented, while remaining positive at both extremes. Because $G = 8$, empirical pass rates occur in 12.5-point increments; the displayed intervals are disjoint reporting bins.

Table 12: Cross-family transfer to Llama-3.1-8B-Instruct without retuning ρ, K, or ϵ_w.

Setting	Backbone	v3 Overall	v3 Multi-turn	v4 Overall	Tau3
GRPO	Qwen3-4B-Thinking-2507	$71.25_{\pm 0.37}$	$47.17_{\pm 0.69}$	$41.25_{\pm 0.26}$	$44.77_{\pm 0.63}$
CFT	Qwen3-4B-Thinking-2507	$74.37_{\pm 0.29}$	$51.41_{\pm 0.36}$	$43.41_{\pm 0.16}$	$53.10_{\pm 0.40}$
GRPO	Llama-3.1-8B-Instruct	$66.82_{\pm 0.44}$	$33.71_{\pm 0.82}$	$37.94_{\pm 0.31}$	$38.15_{\pm 0.71}$
CFT	Llama-3.1-8B-Instruct	$69.53_{\pm 0.33}$	$37.46_{\pm 0.54}$	$39.72_{\pm 0.22}$	$45.28_{\pm 0.49}$

The Llama tokenizer segments tool names into 6.1 subword pieces on average, versus 5.2 under Qwen3 (17% more fragmented). The IG_t selector nevertheless retains a structural precision of 76% (versus 79%), answer-surface overlap stays low (14.2% versus 13.4%), and perturbing selected positions reduces reward by 13.9 / 14.7 / 18.4 on BFCL v3/v4/Tau3. Within each run, student and teacher share the same backbone and tokenizer, so no cross-vocabulary alignment is needed.

Table 13: Transfer to non-tool mathematical RLVR using the Qwen3-4B backbone, CFT hyperparameters, ground-truth final answers as priv, and binary outcome rewards.

Setting	MATH500	AIME25
GRPO	$94.62_{\pm 0.31}$	$82.71_{\pm 0.94}$
CFT (Ours)	$\mathbf{95.47}_{\pm 0.24}$	$\mathbf{84.06}_{\pm 0.81}$
Gain	$+0.85$	$+1.35$

Perturbing selected positions reduces final-answer accuracy by 14.6 pp, compared with 4.8 pp for matched random positions. These math-trained results are distinct from the tool-trained AIME25 OOD check in Appendix G.

Table 14: Privileged-branch controls using gold, mismatched, and self-consistency pseudo-answers.

Privileged source	Gold in branch?	v3 Overall	v4 Overall	Tau3
Student-entropy control	No	$73.41_{\pm 0.20}$	$42.13_{\pm 0.18}$	$46.08_{\pm 0.57}$
Mismatched answer	No	$72.78_{\pm 0.23}$	$41.74_{\pm 0.19}$	$42.87_{\pm 0.66}$
P1: self-consistency pseudo-answer	No	$74.04_{\pm 0.27}$	$42.88_{\pm 0.21}$	$50.46_{\pm 0.51}$
Gold per-step reference	Yes	$\mathbf{74.37}_{\pm 0.29}$	$\mathbf{43.41}_{\pm 0.16}$	$\mathbf{53.10}_{\pm 0.40}$

P1 retains 65.6%/58.6%/62.4% of the student-entropy-to-gold gain on BFCL v3/v4/Tau3. The evaluated structured verifier remains reference-based; P1 only removes gold information from the localization branch.

Table 15: Intervention-path attribution for noisy-answer degradation. N1 uses the mismatched privileged source for selection but the gold source for weighting.

Setting	v3 Overall	v4 Overall	Tau3
CFT: gold selection / gold weighting	$74.37_{\pm 0.29}$	$43.41_{\pm 0.16}$	$53.10_{\pm 0.40}$
N1: mismatched selection / gold weighting	$73.06_{\pm 0.25}$	$42.03_{\pm 0.20}$	$44.39_{\pm 0.58}$
Mismatched selection / mismatched weighting	$72.78_{\pm 0.23}$	$41.74_{\pm 0.19}$	$42.87_{\pm 0.66}$

Along this fixed intervention path, changing selection accounts for 82.4%/82.6%/85.1% of the total degradation on BFCL v3/v4/Tau3. This is a path-specific attribution rather than an order-independent factorial decomposition.

D.3 Compute and Memory Decomposition

Table 16: Measured seconds per full rollout-and-optimization update on 4×A100 GPUs (single profiling run under the configuration of Table 7).

Component	GRPO	CFT	Difference
Rollout generation	122.4	122.4	0.0
Student forward + backward	61.5	61.5	0.0
Answer-conditioned teacher forward	N/A	22.3	+22.3
Entropy + IG_t selection	N/A	0.3	+0.3
Union top-k KL + reweighting	N/A	0.2	+0.2
Optimizer + other	10.5	10.5	0.0
Total	**194.4**	**217.2**	**+22.8 (+11.7%)**

The teacher forward accounts for 97.8% of added time; entropy, selection, KL, and reweighting together cost 0.5 seconds, or 0.23% of one update. The displayed per-update times are rounded; time-to-target uses the observed wall-clock measurements.

Table 17: Measured peak memory per device for the default and exact full-vocabulary KL variants (single profiling run under the configuration of Table 7).

Component	GRPO	CFT	Note
Teacher forward working set	N/A	3.8 GB	logits + KV, no-grad
Entropy + IG_t workspace	N/A	0.6 GB	chunked reduction
Union top-k KL buffers	N/A	0.1 GB	$[T, 2K+1]$, $K = 20$
Peak reserved	**58.4 GB**	**62.9 GB**	**+4.5 GB (+7.7%)**
Exact full-vocabulary KL	N/A	76.2 GB	+17.8 GB (+30.5%)

The union support has at most $2K + 1 = 41$ entries per position, compared with $V = 151{,}936$ vocabulary entries. The coarse-grained KL is therefore a lower-bound approximation to the corresponding full-vocabulary divergence under the stated partition.

E Extended Token Selection, Mask, and KL Ablations

This appendix supplements the token selection, mask design, and KL approximation experiments corresponding to Table 3 and Figure 2a–b in the main text. Unless otherwise noted, all tables report mean(std) over 3 independent training seeds.

Additional controls remove overlap-positive positions (Table 19), compare selector implementations (Table 20), isolate KL-only masking (Table 21), vary mask ratios (Table 22), and compare KL approximations (Table 23).

For a training trajectory with ground-truth answer, let S be the set of positions selected by the selector and $\mathcal{A}(y)$ the token set of the answer. The overlap rate is defined as

$$\text{OverlapRate} = \frac{1}{|S|} \sum_{t \in S} \mathbf{1}[y_t^{(i)} \in \mathcal{A}(y)]. \tag{36}$$

Table 18: Overlap rate between selected positions and answer surface tokens.

Selector	Overlap Rate (%)
$\boldsymbol{IG_t}$	$\mathbf{13.4_{\pm 0.4}}$
student-only entropy	$10.7_{\pm 0.5}$
random	$7.6_{\pm 0.3}$

Table 19: Control experiment removing all overlap-positive positions.

Setting	BFCL v3 Overall	BFCL v4 Overall	Tau3 Overall
IG_t (full)	$74.37_{\pm 0.29}$	$43.41_{\pm 0.16}$	$53.10_{\pm 0.40}$
IG_t w/o overlap-positive positions	$74.13_{\pm 0.25}$	$43.17_{\pm 0.17}$	$52.56_{\pm 0.46}$

Table 20: Stability of the default selector.

Selector Implementation	BFCL v3 Overall	BFCL v4 Overall	Tau3 Overall
batch-global top-20% (Ours)	$\mathbf{74.37_{\pm 0.29}}$	$\mathbf{43.41_{\pm 0.16}}$	$\mathbf{53.10_{\pm 0.40}}$
per-trajectory top-20%	$74.00_{\pm 0.22}$	$43.05_{\pm 0.18}$	$52.39_{\pm 0.48}$

Table 21: Comparison of KL-only mask vs. IG_t mask.

Mask Setting	BFCL v3 Overall	BFCL v4 Overall	Tau3 Overall
KL-only mask [top-20% by KL(S→T)]	$73.31_{\pm 0.22}$	$42.13_{\pm 0.21}$	$46.24_{\pm 0.60}$
IG_t mask (Ours)	$\mathbf{74.37_{\pm 0.29}}$	$\mathbf{43.41_{\pm 0.16}}$	$\mathbf{53.10_{\pm 0.40}}$

Table 22: Mask ratio ablation.

Mask Ratio	BFCL v3 Overall	BFCL v4 Overall	Tau3 Overall
top-10% IG_t mask	$74.05_{\pm 0.23}$	$43.06_{\pm 0.18}$	$51.90_{\pm 0.49}$
top-20% IG_t mask (Ours)	$\mathbf{74.37_{\pm 0.29}}$	$\mathbf{43.41_{\pm 0.16}}$	$\mathbf{53.10_{\pm 0.40}}$
top-30% IG_t mask	$74.14_{\pm 0.21}$	$43.17_{\pm 0.17}$	$52.36_{\pm 0.46}$
bottom-70% IG_t mask	$68.36_{\pm 0.27}$	$39.71_{\pm 0.15}$	$41.88_{\pm 0.67}$

Table 23: KL approximation ablation.

KL Computation	BFCL v3 Overall	BFCL v4 Overall	Tau3 Overall
full-vocab KL	$74.48_{\pm 0.27}$	$43.54_{\pm 0.15}$	$53.72_{\pm 0.36}$
top-k+tail KL (Ours)	$\mathbf{74.37_{\pm 0.29}}$	$\mathbf{43.41_{\pm 0.16}}$	$\mathbf{53.10_{\pm 0.40}}$

Overall, Appendix E and Figure 2a–b in the main text jointly support the same conclusion: CFT identifies privileged-information-conditioned decision forks, not simple answer surface overlap or a fragile selection heuristic.

F Additional Mechanism Visualizations

This appendix provides four independent PDF case studies (Figures 3, 4, 5, and 6) to supplement the mechanism analysis in Figure 2a–b. Each figure shows a complete trajectory fragment, an IG_t gradient heatmap over the final model-output segment, and the corresponding local fork candidates; batch-global top-20% Correctable Fork Tokens are highlighted with deeper shading.

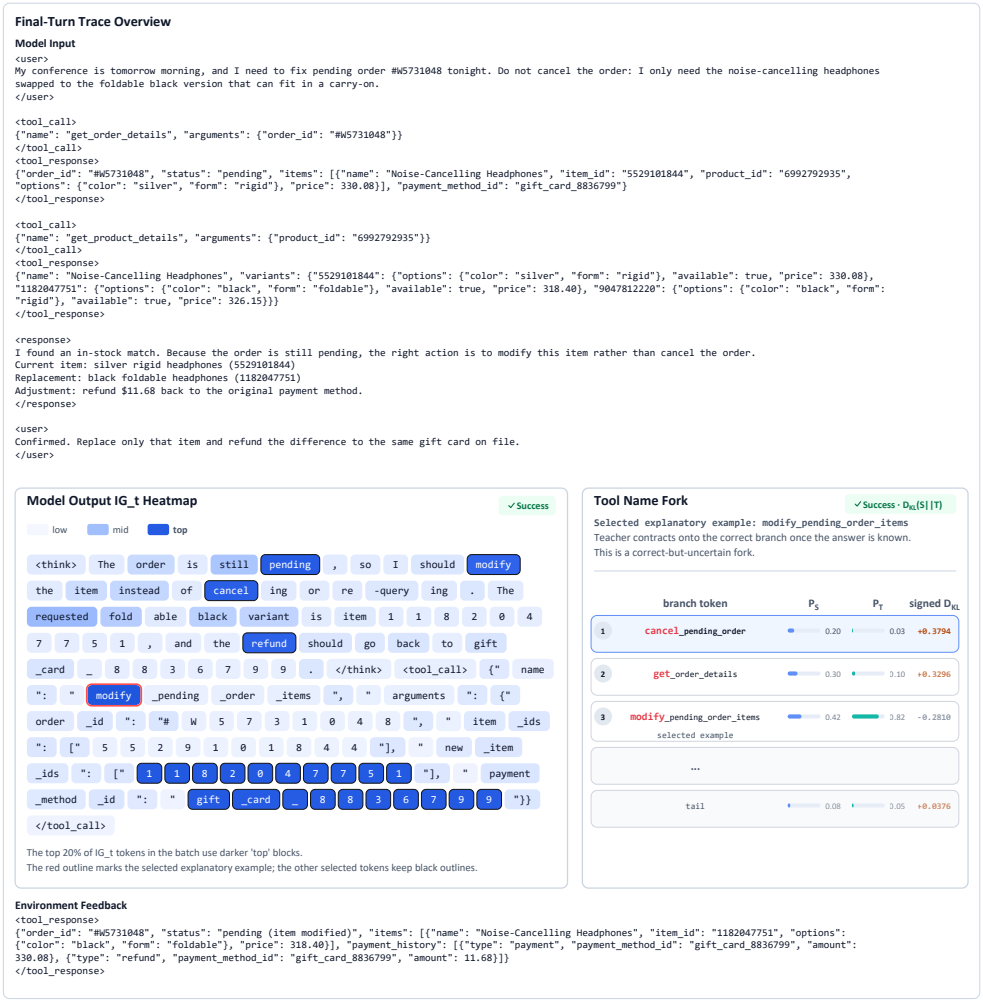


Figure 3: Tool-name fork in a successful trajectory. The student distribution is spread across multiple order operations, while the answer-conditioned teacher concentrates on `modify_pending_order_items`, marking this position as a correct-but-uncertain fork.

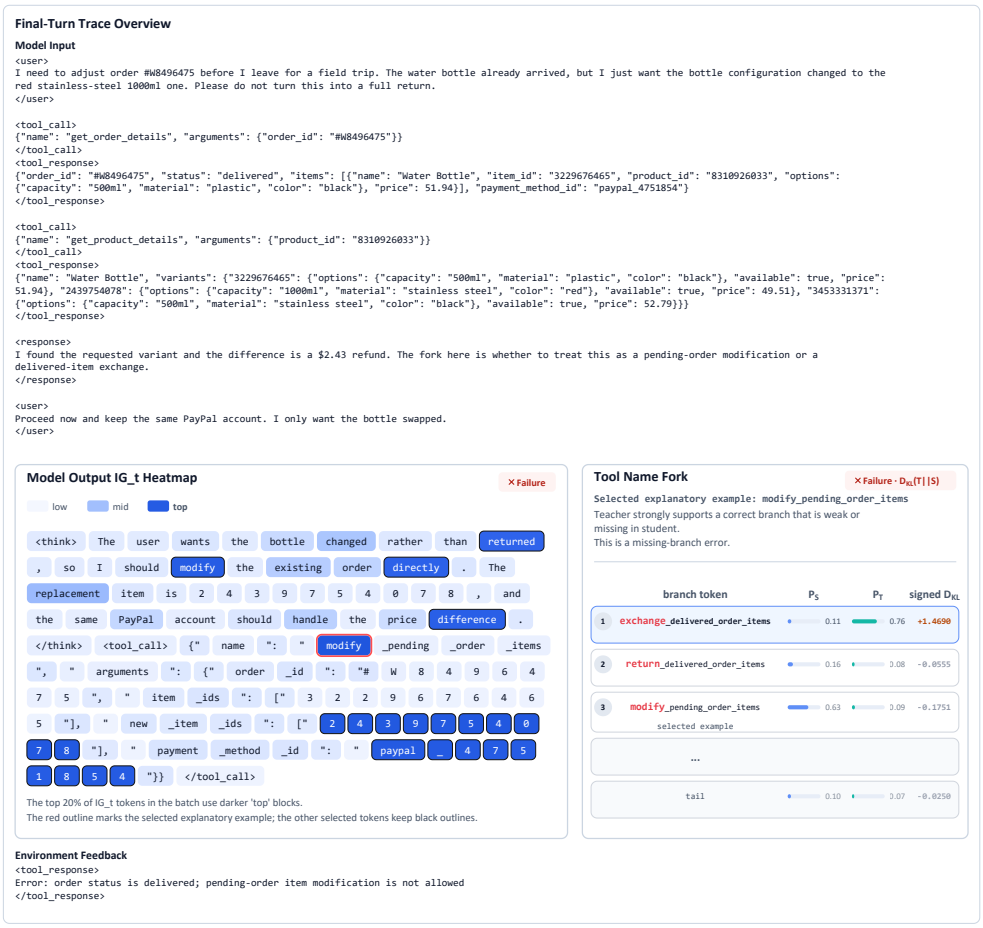


Figure 4: Wrong tool-name fork in a failed trajectory. The student follows `modify_pending_order_items` into the wrong branch, while the teacher concentrates on the correct delivered-order tool—a missing-branch error.



Figure 5: Argument-value fork in a successful trajectory. High-IG_t positions appear at payment method selection, showing that correctable uncertainty persists on critical argument values even when the trajectory ultimately succeeds.

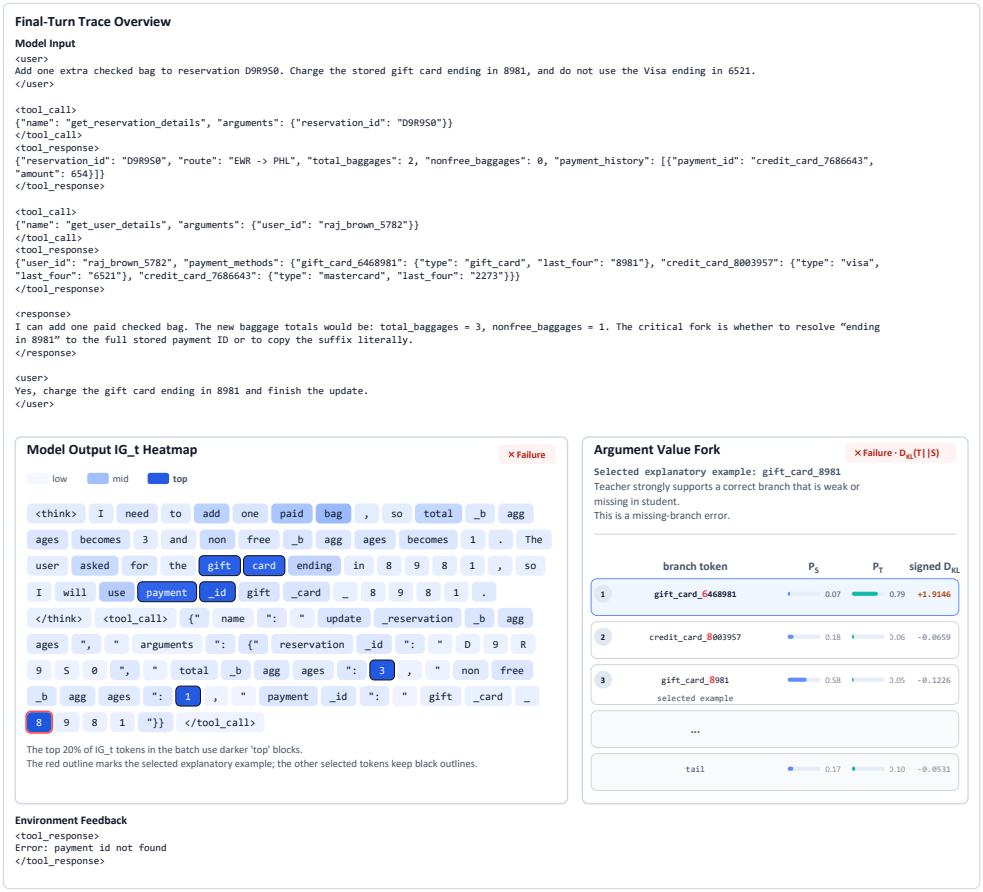


Figure 6: Argument-value error in a failed trajectory. The student uses an abbreviated suffix as the payment ID, while the answer-conditioned teacher concentrates on the full card number—a typical missing-branch parameter error.

G Generalization and OOD

We provide a lightweight generalization/OOD check to verify that tool-reasoning gains are not accompanied by systematic degradation of general capabilities.

Table 24: General capability and OOD check. This table verifies that tool-reasoning gains are not achieved at the cost of systematic degradation in general capabilities.

Benchmark	Capability	Base	Ours	Delta	Note
IFEval	Instruction following	87.4	87.9	+0.5	slightly higher
MMLU-Pro	General knowledge	74.0	74.2	+0.2	largely on par, slightly higher
AIME25	Math reasoning	81.3	80.9	-0.4	slightly lower, but not a systematic drop

Table 24 shows that general capabilities remain stable overall with no systematic degradation: IFEval and MMLU-Pro slightly improve, while AIME25 changes by only -0.4 pp.

H Training Procedure

This appendix provides the procedural description of the structured verifier from Section 3.3 and the optimization procedure from Section 3.4. Algorithm 1 covers the full training loop; Algorithms 2–4 correspond to step-level total reward, tool-call reward, and matched-pair fine scoring, respectively.

Algorithm 1 Correctable Fork Token Training

Require: Policy π_θ, dataset $S = \{(x_i, \text{priv}_i)\}_{i=1}^{N}$, verifier $R(\cdot,\cdot)$, group size G, top-k size K, mask ratio ρ, clip bounds ϵ, ϵ_w

for each training iteration **do**
  Sample a batch of questions $\{x\}$ from S
  // Steps 1–2: On-policy rollout and IG_t collection (per question)
  for each question x with privileged information priv **do**
    Sample G responses $\{y^{(1)}, \ldots, y^{(G)}\} \sim \pi_\theta(\cdot \mid x)$
    for $i = 1, \ldots, G$ **do**
      Obtain reward $R(x, y^{(i)})$ from the verifier ▷ See Algorithm 2
    end for
    for $i = 1, \ldots, G$ **do**
      $A^{(i)} \leftarrow R(x, y^{(i)}) - \frac{1}{G-1} \sum_{j \neq i} R(x, y^{(j)})$
      Run student on $(x, y^{(i)})$ and teacher on $(x, \text{priv}, y^{(i)})$
      for $t = 1, \ldots, |y^{(i)}|$ **do**
        $P_S^t(v) \leftarrow \pi_\theta(v \mid x, y_{<t}^{(i)}), \quad P_T^t(v) \leftarrow \pi_\theta(v \mid x, \text{priv}, y_{<t}^{(i)})$
        $H_S(t) \leftarrow -\sum_v P_S^t(v) \log P_S^t(v), \quad H_T(t \mid \text{priv}) \leftarrow -\sum_v P_T^t(v) \log P_T^t(v)$
        $IG_t^{(i)} \leftarrow H_S(t) - H_T(t \mid \text{priv})$
      end for
    end for
  end for
  // Step 3: Batch-global Correctable Fork Token selection
  $\tau_\rho^{\mathcal{B}} \leftarrow$ top-ρ quantile of $\{IG_t^{(i)}\}$ over all questions in batch ▷ Padding tokens excluded
  // Step 4: Mask and token weighting (per question)
  for each question x **do**
    for $i = 1, \ldots, G$ **do**
      $m_t^{(i)} \leftarrow \mathbf{1}[IG_t^{(i)} \geq \tau_\rho^{\mathcal{B}}]$ for all t; $\quad S^{(i)} \leftarrow \{t : m_t^{(i)} = 1\}$
      for $t = 1, \ldots, |y^{(i)}|$ **do**
        $\mathcal{V}_t^K \leftarrow \text{TopK}(P_S^t) \cup \text{TopK}(P_T^t)$; $\quad$ Build $\tilde{P}_S^t, \tilde{P}_T^t$ on $\mathcal{V}_t^K \cup \{\text{tail}\}$
        if $A^{(i)} > 0$ **then**
          $s_t^{(i)} \leftarrow D_{KL}(\tilde{P}_S^t \,\|\, \tilde{P}_T^t)$
        else if $A^{(i)} < 0$ **then**
          $s_t^{(i)} \leftarrow D_{KL}(\tilde{P}_T^t \,\|\, \tilde{P}_S^t)$
        else
          $s_t^{(i)} \leftarrow 0$
        end if
      end for
      if $\sum_{j \in S^{(i)}} s_j^{(i)} = 0$ **then**
        Set $w_t^{(i)} \leftarrow 1$ for all t
      else
        $w_t^{(i)} \leftarrow \dfrac{s_t^{(i)}}{\frac{1}{|S^{(i)}|} \sum_{j \in S^{(i)}} s_j^{(i)}}$
      end if
      $\bar{A}_t^{(i)} \leftarrow \text{clip}(w_t^{(i)}, 1-\epsilon_w, 1+\epsilon_w) \cdot A^{(i)}$
    end for
  end for
  // Step 5: Masked clipped surrogate objective
  $r_t^{(i)}(\theta) \leftarrow \dfrac{\pi_\theta(y_t^{(i)} \mid x, y_{<t}^{(i)})}{\pi_{\theta_{\text{old}}}(y_t^{(i)} \mid x, y_{<t}^{(i)})}$

$$\mathcal{L}_{\text{CFT}}(\theta) \leftarrow -\mathbb{E}\left[\frac{1}{\sum_{i=1}^{G}\sum_{t=1}^{|y^{(i)}|} m_t^{(i)}} \sum_{i=1}^{G}\sum_{t=1}^{|y^{(i)}|} m_t^{(i)} \min\Big(r_t^{(i)}(\theta)\bar{A}_t^{(i)}, \text{clip}(r_t^{(i)}(\theta), 1-\epsilon, 1+\epsilon)\bar{A}_t^{(i)}\Big)\right]$$

  Update θ using $\mathcal{L}_{\text{CFT}}(\theta)$
end for

Algorithm 2 Total Reward Calculation

function CALCULATETOTALREWARD(Generation, GroundTruth, C)
Input:
Generation: model output string.
GroundTruth: reference label string.
C: correctness weight.
Output:
R: reward score in $[-C, 1 + C]$.
$R_{\text{format}} \leftarrow$ FORMATINDICATOR(Generation) ▷ Valid structured pattern from Section 3.3
$P_{\text{calls}} \leftarrow$ PARSETOOLCALLS(Generation)
$G_{\text{calls}} \leftarrow$ PARSETOOLCALLS(GroundTruth)
$R_{\text{correct}} \leftarrow$ CALCULATETOOLREWARD($P_{\text{calls}}, G_{\text{calls}}, C$) ▷ See Algorithm 3
$R \leftarrow R_{\text{format}} + R_{\text{correct}}$
return R
end function

Algorithm 3 Tool Call Reward Calculation

function CALCULATETOOLREWARD($P_{\text{calls}}, G_{\text{calls}}, C$)
$m \leftarrow |P_{\text{calls}}|,\ n \leftarrow |G_{\text{calls}}|$
$M \leftarrow \varnothing,\ U \leftarrow G_{\text{calls}},\ \text{total_score} \leftarrow 0$
for each predicted call $p_i \in P_{\text{calls}}$ **do**
best_overlap $\leftarrow -1,$ best_match $\leftarrow$ null
for each reference call $g_j \in U$ **do**
if p_i.name $= g_j$.name **then**
$K_p \leftarrow \text{keys}(p_i.\text{arguments}),\ K_g \leftarrow \text{keys}(g_j.\text{arguments})$
if $K_p = K_g = \varnothing$ **then** ▷ Handle empty argument lists to avoid division by zero
overlap $\leftarrow 1$
else
overlap $\leftarrow |K_p \cap K_g|/|K_p \cup K_g|$
end if
if overlap $>$ best_overlap **then**
best_overlap $\leftarrow$ overlap, best_match $\leftarrow g_j$
end if
end if
end for
if best_match $\neq$ null **then**
$M \leftarrow M \cup \{(p_i, \text{best_match})\}$
total_score $\leftarrow$ total_score + MATCHEDPAIRFINESCORE(p_i, best_match) ▷ See Algorithm 4
Remove best_match from U
end if
end for
if $m = 0$ and $n = 0$ **then**
$R_{\text{Tools}}^{\text{fine}} \leftarrow 3$
else
$R_{\text{Tools}}^{\text{fine}} \leftarrow \text{total_score}/(m + n - |M|)$
end if
return $C \cdot (2 \cdot R_{\text{Tools}}^{\text{fine}}/3 - 1)$
end function

Algorithm 4 Matched-Pair Fine Scoring

function MATCHEDPAIRFINESCORE(p, g)
$K_p \leftarrow \text{keys}(p.\text{arguments}),\ K_g \leftarrow \text{keys}(g.\text{arguments})$
if $K_p = K_g = \varnothing$ **then**
$r_{\text{argname}} \leftarrow 1,\ r_{\text{argvalue}} \leftarrow 1$
else
$r_{\text{argname}} \leftarrow |K_p \cap K_g|/|K_p \cup K_g|$
value_sum $\leftarrow 0$
for each key $k \in K_p \cap K_g$ **do**
$p_k \leftarrow p.\text{arguments}[k],\ g_k \leftarrow g.\text{arguments}[k]$
if p_k, g_k are strings **then**
$\text{score}_k \leftarrow$ ROUGE-L-F1(p_k, g_k)
else if p_k, g_k are numeric/boolean **then**
$\text{score}_k \leftarrow (1$ if $p_k = g_k$ else $0)$
else
$\text{score}_k \leftarrow (1$ if canonical_str$(p_k) =$ canonical_str(g_k) else $0)$ ▷ Sort dict keys before serializing nested structures
end if
value_sum $\leftarrow$ value_sum + score_k
end for
$r_{\text{argvalue}} \leftarrow \text{value_sum}/|K_p \cup K_g|$
end if
return $1 + r_{\text{argname}} + r_{\text{argvalue}}$
end function

The constant 1 corresponds to a correct tool match verified by call-level alignment.

I Prompt Templates and Input Construction

This appendix supplements Section 3.2 with input templates and compressed-history diagrams (Figures 7, 8, and 9) to illustrate the structured rollout format, teacher input construction, and how multi-turn history is retained in context.

System Prompt

You are a helpful multi-turn dialogue assistant capable of leveraging tool calls to solve user tasks and provide structured chat responses.

Tools

You may call one or more functions to assist with the user query.

You are provided with function signatures within <tools></tools> XML tags:
<tools>
{{Tool List}}
</tools>

For each function call, return a json object with function name and arguments within <tool_call></tool_call> XML tags:
<tool_call>
{\"name\": <function-name>, \"parameters\": <args-json-object>}
</tool_call>

Available Output Format

<think>
Your thoughts and reasoning
</think>

<tool_call>
{\"name\": \"Tool name\", \"arguments\": {\"arguments name\": \"arguments content\", \"... ...\": \"... ...\"}}
</tool_call>

<response>
AI's response
</response>

Important Notes

1. **Think:** Always recall relevant context and analyze the current user goal. Include a <think> field for your reasoning, and reference dialogue history — including user queries, previous <tool_call>, <response>, and <tool_response>.
2. Choose either 'Respond' or 'Tool Usage' — not both!
3. **Tool Usage:** If a tool is needed, use the <tool_call> field. Each call must be a JSON object with \"name\" and \"arguments\". If no parameters are needed, leave the "parameters" field an empty dictionary. If no tool can be used, or if the question lacks required parameters, explicitly point it out.
4. **Respond:** If a response is needed, use the <response> field and generate one while maintaining consistency across user queries.

Figure 7: The system prompt defines the structured rollout format and available tool set for each step.

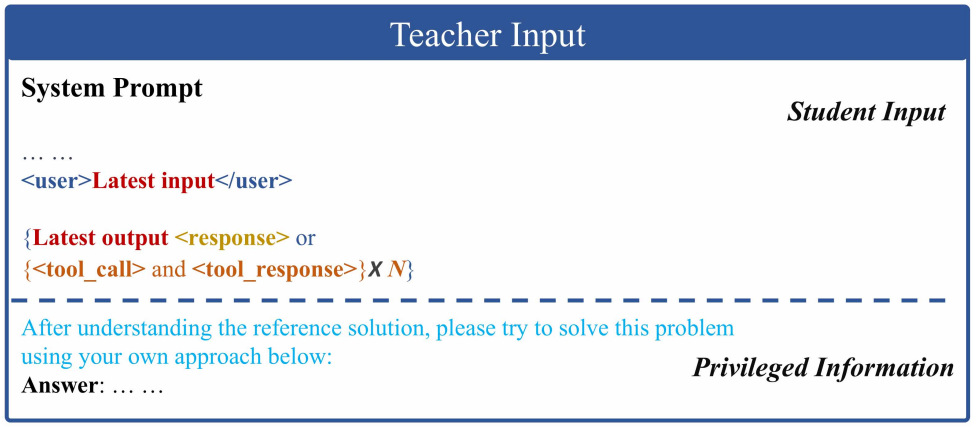


Figure 8: The teacher input appends the training-time per-step reference output to the student input. This reference is used for credit localization, not as a token-level imitation target.

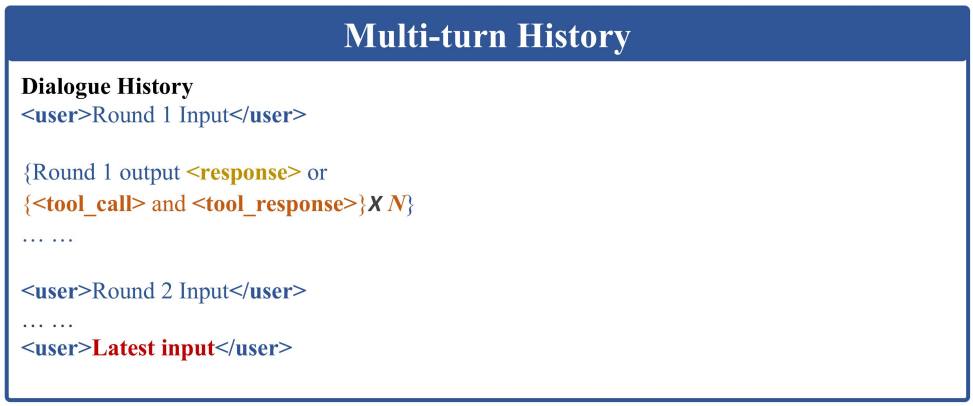


Figure 9: The compressed multi-turn history retains only decision-relevant context in the model input.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Introduction state the method, assumptions, and empirical claims, and Sections 4.2–4.6 provide the corresponding BFCL, Tau3, ablation, mechanism, and OOD evidence. The stated scope is limited to tool-integrated RLVR with training-time privileged answer information, as discussed in Sections 3.1 and 6.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 6 discusses reliance on training-time privileged signals, the current model-family scope, the open problem of confidently wrong decisions, and deployment-related risks for tool-use systems. The compute and memory overhead of the answer-conditioned branch is quantified in Table 7 and Appendix Tables 16–17.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Appendix B states the assumptions for the structural theoretical results and provides complete proofs. The results are limited to direction preservation, credit concentration under high-precision selection, the expected mutual-information interpretation of IG_t under calibration, and the conservative nature of the top-k+tail KL approximation.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 4.1 specifies the datasets, model backbones, training settings, evaluation settings, baselines, and aggregation procedure. Appendix H gives the training and verifier algorithms, and Appendix I documents the prompt/input construction.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The abstract links to the public GitHub repository `https://github.com/YSQ-boop/CFT`, which provides the implementation, configurations, and reproduction scripts. The training data, evaluation benchmarks, and model backbones are existing cited assets described in Section 4.1.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1 gives the training data construction, batch size, rollout group size, temperature, number of epochs, KL coefficient, CFT weight clip, reward setting, evaluation benchmarks, simulator settings, and baseline definitions. Appendix H and Appendix I provide implementation-level procedure and input-format details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: Section 4.1 states that results are reported as mean(std) over three independent training seeds except for explicitly noted tables. For Tau3, each seed averages pass@1 over three runs per task before aggregation across seeds.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Table 7 reports time-to-target, wall-clock time per update, and updates-to-target on 4×A100 GPUs. Appendix Tables 16 and 17 decompose per-update time and peak memory per device, including the extra answer-conditioned teacher forward.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses existing datasets, benchmarks, and models with cited sources. It does not involve human-subject experiments or crowdsourcing, and the broader-impact risks are discussed in Section 6.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 6 discusses the intended positive impact of more reliable tool-integrated LLM training and the negative risk that improved tool use could amplify automation failures or misuse without downstream safeguards.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a new high-risk pretrained model, image generator, or scraped dataset. Responsible deployment considerations for tool-use systems are discussed in Section 6.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Section 4.1 credits the existing training data, evaluation benchmarks, and model backbones through citations and model names. All assets are publicly available and are used under their original licenses and terms of use; the public code repository lists each asset's source, version, and license, together with access and preparation instructions.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The released assets are the implementation, configurations, data-processing instructions, prompt templates, and experiment scripts, available in the public repository linked from the abstract. The paper does not introduce a new benchmark dataset or release a new pretrained foundation model.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The research does not involve crowdsourcing, human-subject experiments, or participant compensation. It uses existing datasets, benchmarks, automated training, and automated evaluation.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human-subject research or crowdsourcing is conducted, so IRB or equivalent approval is not applicable. The experiments use existing machine-learning datasets and automated benchmark environments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are the central research object and method component: Sections 3.1–3.4 define the student and same-parameter answer-conditioned teacher branches, and Section 4.1 specifies the Qwen3-family [27] student, teacher, expert, and simulator models used in experiments.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.