TCCA: TARGET-CONDITIONED CREDIT ASSIGNMENT FOR TEST-TIME REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time reinforcement learning derives supervision from self-generated answers, but outcome-only credit loses information at two levels: within a prompt, responses with the same binary reward receive identical base advantages, while across prompts, within-prompt normalization does not reflect differences in target ambiguity. We introduce Target-Conditioned Credit Assignment (TCCA), which keeps a single hard pseudo-target estimated from the rollout pool and uses it to organize learning credit at two complementary resolutions. Self-certainty-weighted Soft Vote estimates the target and its answer distribution. The associated ambiguity reallocates normalized outcome credit across prompts through Hardness-Oriented Advantage Scaling, while target-matching reasoning traces form compact semantic references that provide relative reasoning credit within each prompt. The two terms are normalized separately and combined at the advantage stage for the same policy update, reusing the sampled responses and a frozen semantic encoder without a dedicated process reward model. Across single-run evaluations on three backbones and four reasoning benchmarks, TCCA yields higher point estimates than outcome-only TTRL in all 12 settings. In matched three-seed MATH-500 experiments, TCCA improves sample accuracy / majority correctness from 38.96/40.40 to 41.10/42.80 on Qwen2.5-0.5B and from 39.80/41.73 to 42.82/44.67 on Llama3.2-1B-Instruct, with positive paired differences in all three seeds on both backbones. Component removals and a separate one-versus-two-prototype comparison further support the credit construction, with approximately 5.3% additional complete-run time on the measured setup.

1 INTRODUCTION

Test-time reinforcement learning (TTRL) adapts a language model on unlabeled test inputs using supervision derived from its own responses (Zuo et al., 2025). A policy samples several reasoning traces, selects a pseudo-target through answer agreement, and receives binary rewards for matching that target. This extends the use of self-consistency from answer selection (Wang et al., 2022) to policy learning. It provides an opportunity to improve reasoning without task-specific training labels, while making the construction of self-generated supervision central to the update.

Binary outcome rewards compress the information in these responses. Within a prompt, trajectories with the same reward receive the same base advantage after group normalization, despite differences in their reasoning. Across prompts, answer distributions can also exhibit different degrees of ambiguity. The rollout pool contains both the reasoning traces and the answer agreement needed to address these limitations. We therefore ask: *how can this information enrich learning credit beyond the binary match decision?*

We introduce *Target-Conditioned Credit Assignment* (TCCA). Soft Vote uses self-certainty-weighted voting to estimate one hard pseudo-target and its answer distribution. The target is shared across the two credit pathways: the target defines binary outcome supervision, while Hardness-Oriented Advantage Scaling (HOAS) uses ambiguity in the associated distribution to weight normalized outcome advantages across a prompt batch. The same target selects the reasoning references used to construct additional credit within each prompt. This semantic *reasoning credit* provides prompt-local relative comparisons among traces; it is not intended as a verifier of deductive validity.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

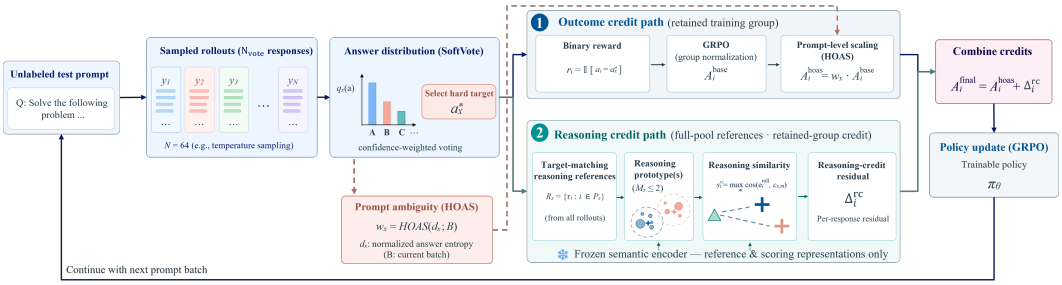


Figure 1: **One target organizes two credit pathways.** Soft Vote estimates $a_x^\star$ and q_x from the full 64-response pool. The target defines binary outcome rewards and selects reasoning references; answer ambiguity controls HOAS. Reasoning similarities are converted to residuals over the retained group T_x, and both credit terms enter the same policy update. Here P_x contains target-matching indices and $R_x = \{\tau_i : i \in P_x\}$ contains their reasoning traces. The first 32 responses form the retained training group T_x.

TCCA constructs this second pathway from the existing rollout pool. A frozen semantic encoder represents target-matching reference traces, and conservative clustering summarizes supported regions with at most two prototypes. All scoreable retained responses are compared with these references, including retained responses whose answers differ from the target. Their similarities are converted into a separately normalized reasoning residual. No dedicated process reward model is trained for this construction. Outcome weighting and reasoning credit are combined at the advantage stage, as shown in Figure 1.

We evaluate TCCA from broad coverage to repeated comparisons. Single-run experiments yield higher point estimates than Original TTRL across all 12 model–benchmark combinations. Under matched three-seed MATH-500 protocols, TCCA improves Qwen2.5-0.5B by 2.14 sample-accuracy points and 2.40 majority-correctness points, and Llama3.2-1B-Instruct by 3.02 and 2.93 points, respectively, with positive paired differences in all three seeds for both backbones. On the Qwen setting, TCCA also exceeds the reproduced three-seed means of SCOPE by 3.57/3.93 points and ETTRL Adv-Res by 1.06/1.20 points in sample accuracy / majority correctness. Removing reasoning credit, HOAS, or Soft Vote lowers mean sample accuracy by 1.33, 1.18, and 1.40 points, respectively, while a separate paired comparison favors at most two prototypes over one by 1.74 points. The complete implementation adds approximately 5.3% elapsed time on the measured setup. Fixed-pool analyses separately characterize what additional credit the method introduces.

Our contributions are threefold:

- We formulate target-conditioned credit assignment around a single hard pseudo-target estimated from the rollout pool. Its answer distribution provides an across-prompt ambiguity signal, while its target-matching traces define within-prompt semantic references.
- We realize these two resolutions through post-normalization HOAS and a separately normalized prototype-based reasoning residual, then combine them directly at the advantage stage without training a task-specific process reward model.
- We evaluate TCCA across three backbones and four benchmarks, together with matched three-seed validation on two backbones, component removals, prototype-count comparisons, fixed-pool diagnostics, and matched-hardware runtime measurements.

2 Related Work

Self-generated targets and prompt-level allocation. TTRL derives binary supervision from a majority-vote pseudo-target on unlabeled inputs (Zuo et al., 2025). SCOPE constructs subgroup-specific targets using step-wise confidence (Wang et al., 2026), while SCRL combines selective positive and complementary negative pseudo-labels (Yan et al., 2026); these methods primarily modify which answers provide supervision. Other work instead changes how uncertain self-supervision contributes to optimization. ETTRL couples entropy-guided tree sampling with advantage reshaping

(Liu et al., 2025), SEED-GRPO attenuates updates on high-semantic-entropy prompts (Chen et al., 2025), DDRL filters ambiguous samples and debiases group-relative advantages (Yu et al., 2026), and RESTRAIN uses self-penalization to reduce reliance on spurious self-generated signals (Yu et al., 2025). These methods generally reduce the influence of uncertain supervision through filtering, attenuation, or penalization. TCCA addresses a different role of uncertainty: Soft Vote first selects one hard target, after which the associated answer ambiguity is used only as a bounded, batch-relative signal to reallocate already normalized outcome credit across prompts. Thus, ambiguity is not treated as a certificate of target reliability or as a standalone reward. Section 4.1 specifies the shared-trainer SCOPE and ETTRL advantage-reshaping comparisons.

Within-prompt reasoning credit and reference geometry. COMPASS combines a confidence-weighted answer target with answer and path rewards based on token-distribution decisiveness (Xing et al., 2025), while CoRE jointly uses answer agreement, reasoning similarity and confidence to construct refined supervision (Mehrish & Vascon, 2026). ReasonFlux-PRM instead uses learned step- and trajectory-level supervision (Zou et al., 2025). Geometric approaches include Latent-GRPO, which estimates a robust centroid from policy hidden states (Zhang et al., 2026), and RCS, which uses an answer-embedding center for best-of-N selection (Nguyen et al., 2026). TCCA conditions its geometry on the selected target: only target-matching traces define the semantic reference space, while retained responses, including nonmatching ones, are scored relative to that space. This restores within-prompt distinctions beyond binary outcome credit without training a task-specific process reward model.

Credit integration and positioning. SPO obtains finer credit through Monte Carlo continuations (Guo et al., 2025), whereas TCCA reuses the parallel rollout pool. PRPO converts process-model scores into token-level advantages (Ding et al., 2026), and GDPO separately normalizes heterogeneous reward channels before aggregation (Liu et al., 2026). TCCA differs from generic multi-reward aggregation in that both credit terms arise from one selected target at two distinct resolutions: the answer distribution controls outcome allocation across prompts, while target-matching traces define reasoning distinctions within a prompt. The two terms are normalized separately and combined only at the advantage stage, a design related to separate-channel normalization in GDPO but not to its complete normalization procedure.

3 Target-Conditioned Credit Assignment

TCCA retains the TTRL backbone and separates the construction into two stages: target estimation and target-conditioned credit assignment. Soft Vote first estimates one hard pseudo-target and its answer distribution from the rollout pool. Conditioned on these outputs, TCCA assigns credit at two resolutions: outcome allocation across prompts and reasoning differentiation within each prompt. The two terms are combined after their separate normalizations.

3.1 Confidence-Weighted Target Estimation

For each prompt x, policy π_θ samples $N_{\text{vote}} = 64$ responses. Response i contains an extracted answer label a_i after task-specific answer extraction and normalization, together with a reasoning trace τ_i. All $N_{\text{vote}} = 64$ responses are used for target estimation and reference construction. Following TTRL's vote-then-sample strategy, the first $N_{\text{train}} = 32$ responses in generation order form the retained training group $T_x = \{1, \ldots, N_{\text{train}}\}$ for policy optimization.

Soft Vote uses rollout-level self-certainty $s_i^{\text{sc}} = L_i^{-1} \sum_{t=1}^{L_i} D_{\text{KL}}(U_V \| p_{i,t})$ where L_i is the response length, as a relative reliability signal (Kang et al., 2025; Zhao et al., 2026). The voting weights are $\omega_i = s_i^{\text{sc}} / \sum_{j=1}^{N_{\text{vote}}} s_j^{\text{sc}}$. The answer distribution and selected target are

$$q_x(a) = \sum_{i=1}^{N_{\text{vote}}} \omega_i \mathbb{I}[a_i = a], \qquad a_x^\star = \arg\max_a q_x(a). \tag{1}$$

Here a ranges over the normalized extracted answer labels. The selected target determines outcome rewards and reasoning references, while q_x supplies the ambiguity signal for HOAS. Appendix A.1 defines self-certainty.

3.2 TARGET-CONDITIONED OUTCOME CREDIT

The binary reward is $r_i = \mathbb{I}[a_i = a_x^\star]$. GRPO (Shao et al., 2024) normalizes these rewards within T_x to obtain A_i^{base}. This provides relative outcome credit within a prompt, but does not encode how unresolved the sampled answer distribution remains across prompts. HOAS therefore uses ambiguity in q_x as a bounded, batch-relative allocation signal. Importantly, high ambiguity is not interpreted as evidence that the selected target is reliable: it may reflect either informative competition among sampled answers or noisy self-supervision. We therefore restrict its effect through the raw weight clamp $[0.8, 1.2]$ and batch-mean normalization, rather than using ambiguity as an additional reward or an unbounded confidence signal. Let $\mathcal{A}_x = \{a : q_x(a) > 0\}$ and $K_x = \max(2, |\mathcal{A}_x|)$. Normalized answer entropy is $d_x = -\sum_{a \in \mathcal{A}_x} q_x(a) \log q_x(a) / \log K_x$; a single supported label therefore gives $d_x = 0$. With $\bar{d} = |B|^{-1} \sum_{x \in B} d_x$, define

$$\widetilde{w}_x = \text{clip}\big(1 + \beta(d_x - \bar{d}), w_{\min}, w_{\max}\big), \qquad w_x = \frac{\widetilde{w}_x}{|B|^{-1} \sum_{x' \in B} \widetilde{w}_{x'}}. \tag{2}$$

We use $\beta = 0.5$ and $[w_{\min}, w_{\max}] = [0.8, 1.2]$. HOAS acts after within-prompt normalization because a common positive prompt-level rescaling of the raw rewards would cancel under exact within-group standardization: $A_i^{\text{hoas}} = w_x A_i^{\text{base}}$. It changes the relative emphasis on outcome supervision without changing the target or reward labels. Appendix A.2 gives the details.

3.3 TARGET-CONDITIONED REASONING CREDIT

The same target selects the reference indices $P_x = \{i \in \{1, \ldots, N_{\text{vote}}\} : a_i = a_x^\star\}$. For these responses, reference cleaning removes final-answer spans and lightweight non-reasoning artifacts. A less aggressively cleaned scoring view is constructed for each retained response to be scored, including retained responses whose answers do not match the target.

A frozen Qwen3-Embedding-0.6B encoder (Zhang et al., 2025), denoted by Enc, maps the reference and scoring views to unit embeddings e_i^{ref} and e_i^{roll}, respectively. Reference-set self-certainty is min–max normalized and then converted to weights ν_i for refinement, split support and prototype construction. Conservative clustering retains a second reference group only when sufficient support and separation are present; otherwise it uses one refined group.

Each retained group produces a confidence-weighted unit prototype $c_{x,m}$. When usable references are available, $M_x \in \{1, 2\}$, and response i receives the similarity score

$$s_i^{\text{rc}} = \max_{m=1,\ldots,M_x} \cos(e_i^{\text{roll}}, c_{x,m}). \tag{3}$$

The nearest-prototype rule preserves separate comparison directions when the references support two regions. Finite scores within T_x are standardized, clipped and scaled to obtain Δ_i^{rc}. Normalization uses the retained group, not separate reward classes, so the residual can distinguish responses sharing the same binary reward. This is relative semantic feedback, not a ground-truth verifier of reasoning. Appendices A.3 and A.4 specify the construction and residual.

3.4 CREDIT INTEGRATION AND POLICY UPDATE

Outcome supervision and reasoning feedback enter the same update:

$$A_i^{\text{final}} = A_i^{\text{hoas}} + \Delta_i^{\text{rc}}, \qquad i \in T_x. \tag{4}$$

HOAS weights only the outcome branch. The reasoning residual is added after its own normalization, without jointly renormalizing the sum. This follows the motivation for separate reward-channel normalization (Liu et al., 2026), rather than adopting that method's complete normalization procedure. Varying reasoning scores can provide additional credit even when outcome rewards are homogeneous. The resulting advantages enter the inherited GRPO update with KL regularization. Appendix A.5 explains the normalization argument, and Algorithm 1 summarizes the procedure.

Algorithm 1 TCCA policy update

Input: policy π_θ, prompt batch B, and frozen encoder Enc.
1 For each prompt, sample $N_{\mathrm{vote}} = 64$ responses.
2 Extract answer labels and reasoning traces; compute s_i^{sc}, ω_i, q_x, and $a_x^\star$ from the full rollout pool.
3 Select target-matching references P_x; construct, refine and conservatively group their representations into at most two prototypes.
4 Retain the first $N_{\mathrm{train}} = 32$ responses in generation order as T_x for policy optimization.
5 Compute binary rewards and A_i^{base} over T_x; use q_x to obtain the batch-relative HOAS weight w_x.
6 Score retained responses against their nearest prototype and obtain Δ_i^{rc} after prompt-relative normalization, clipping and scaling.
7 Form A_i^{final} using Eq. 4 and apply the inherited GRPO update.

4 EXPERIMENTS

We first assess broad coverage through single-run experiments, then use paired training runs to evaluate repeatability and compare credit configurations. Fixed-pool analyses characterize the resulting credit signals, and matched-hardware timing measures computational cost.

4.1 EXPERIMENTAL SETUP

We follow TTRL (Zuo et al., 2025) in both the label-free adaptation setting and the training and evaluation recipe. Each benchmark's inputs serve as unlabeled adaptation prompts and as the final evaluation set; gold answers are used only for evaluation and offline analysis. The inherited settings include AdamW with peak learning rate 5×10^{-7} and cosine decay, 64 sampled responses, with the first 32 in generation order retained for optimization, and maximum response length 3072. Evaluation draws 16 responses per prompt at temperature 0.6 and top-$p = 0.95$. Sample accuracy averages response correctness, estimating pass@1 under this decoder; majority correctness evaluates the count-majority answer for each prompt. Table 8 consolidates the shared settings and benchmark schedules.

The broad experiments use single runs without matched training seeds. For repeated validation, we fix training seeds indexed as Seed 1–3. The primary paired setting uses matched evaluation-seed settings across methods and is Qwen2.5-0.5B (Qwen et al., 2025) on MATH-500, containing 500 problems (Hendrycks et al., 2021). All paired methods use 150 optimizer steps and prompt batches of 32 on 8×A6000. We report mean ± sample SD across training runs; Appendix D provides individual scores and paired differences. TCCA hyperparameters are selected during exploratory development and then held fixed for the reported single-run and paired evaluations, without retuning for either backbone setting or individual seeds. Named ablations change only the stated component or prototype cap. Table 7 lists the principal fixed settings.

Reproduced comparisons. Original TTRL (Zuo et al., 2025) assigns binary rewards from the count-majority answer. SCOPE (Wang et al., 2026) instead uses step-wise confidence to form subgroup-specific targets and differentiated rewards. We reproduce its complete target and reward construction from released code on the shared 64-response pool, retaining the first 32 responses for training; its released pipeline optimizes all 64. ETTRL (Liu et al., 2025) combines entropy-guided tree sampling with Entropy-based Advantage Reshaping (EAR). Its branching and continuation procedure differs substantially from the parallel rollouts used by TTRL and our method. An end-to-end comparison would therefore change both the sampled trajectories and the credit rule, and no official implementation was available at reproduction time. We evaluate EAR inside the common parallel trainer as *ETTRL Adv-Res*. Holding the backbone, inputs, rollout budget and optimizer schedule fixed makes this a controlled comparison of credit and advantage construction; it does not represent a full reproduction of ETTRL's tree-search pipeline.

Reported external comparisons. We retain overlapping pass@1 results from COMPASS (Xing et al., 2025), SCRL (Yan et al., 2026) and CoRE (Mehrish & Vascon, 2026). These use the same named backbones and tasks but their own protocols. COMPASS uses roughly 20% of TTRL's epoch budget on Qwen2.5-7B and 30 epochs on Llama/AMC, compared with our 15. SCRL's eligible

rows use Llama3.2-1B-Instruct, 64/32 sampled/retained responses and 16-sample evaluation; its Qwen2.5-Math-7B results are not assigned to vanilla Qwen2.5-7B. CoRE reports three-seed means for vanilla Qwen2.5-7B. These reported values complement the common-trainer experiments.

4.2 Performance across backbones and benchmarks

The broad experiments cover Llama3.2-1B-Instruct (Grattafiori et al., 2024), Qwen2.5-0.5B and Qwen2.5-7B (Qwen et al., 2025) on MATH-500 (Hendrycks et al., 2021), AIME24 and AMC from the TTRL benchmark suite (Zuo et al., 2025), and GPQA-Diamond (Rein et al., 2024). TCCA yields higher single-run point estimates than Original TTRL in all 12 model–benchmark combinations (Table 1). The reported-result comparison places these results in the context of related self-supervised methods: TCCA is higher on six of eight overlapping COMPASS entries, and the SCRL and CoRE comparisons show complementary strengths across tasks. For example, TCCA reaches 43.55% on Llama/MATH-500 against SCRL's 39.70%, while SCRL is higher on AIME24; on Qwen7B, TCCA leads CoRE on MATH-500 and GPQA-Diamond, while CoRE leads on AIME24 and AMC.

Table 1: **Broader reasoning results: pass@1 (%).** Baselines: Original TTRL (Zuo et al., 2025), COMPASS (Xing et al., 2025), SCRL (Yan et al., 2026) and CoRE (Mehrish & Vascon, 2026). Our single-run experiments use independent training seeds without paired matching. Reported rows follow the cited works' respective protocols; CoRE reports three-run means. Shared backbones and tasks have different training budgets (Section 4.1). A dash denotes no eligible reported result.

Backbone	Method	MATH-500	AIME24	AMC	GPQA-D
Llama3.2-1B-Instruct	Base	14.48	0.83	5.65	24.46
	Original TTRL	40.73	0.63	9.49	27.60
	COMPASS (reported)	28.70	3.50	20.10	25.80
	SCRL (ACL 2026)	39.70	6.70	–	–
	TCCA (ours)	43.55	6.04	14.76	27.86
Qwen2.5-0.5B	Base	13.94	0.20	4.67	15.93
	Original TTRL	36.81	0.00	10.62	20.84
	TCCA (ours)	42.00	3.54	13.93	27.98
Qwen2.5-7B	Base	58.43	7.70	33.74	27.57
	Original TTRL	79.63	17.29	52.79	32.07
	COMPASS (reported)	76.90	23.50	53.20	31.70
	CoRE (reported mean)	79.70	23.30	55.40	29.50
	TCCA (ours)	80.50	21.04	54.07	33.19

4.3 From Outcome Reward to Target-Conditioned Credit

The single-run configuration path progresses from outcome-only TTRL to Soft Vote and HOAS, then adds reasoning credit and a second supported prototype. The corresponding scores are 36.81%, 40.71%, 41.06%, and 42.00% (Appendix B, Table 9). These exploratory comparisons motivate the construction; the paired experiments below test repeatability under matched training seeds.

4.4 Paired Multi-Seed Performance

Table 2 evaluates repeatability on MATH-500 across two backbones. On Qwen2.5-0.5B, TCCA improves sample accuracy by 2.14 points and majority correctness by 2.40 points, with positive differences in all three matched pairs. It also has higher three-seed means than SCOPE by 3.57/3.93 points and ETTRL Adv-Res by 1.06/1.20 points in sample accuracy / majority correctness, respectively.

On Llama3.2-1B-Instruct, TCCA increases mean sample accuracy from 39.80 ± 3.04 to 42.82 ± 0.26 and majority correctness from 41.73 ± 3.29 to 44.67 ± 0.12, corresponding to gains of 3.02 and 2.93 points. The paired differences are positive for all three training seeds on both metrics. Appendix D reports the per-seed endpoints and paired differences.

Table 2: **Repeated MATH-500 results across two backbones.** Percentages are mean ± sample SD across three matched training seeds at step 150. Per-seed endpoints and paired differences are reported in Appendix D.

Backbone	Method	Sample accuracy	Majority correctness
Qwen2.5-0.5B	Original TTRL	38.96 ± 0.69	40.40 ± 0.72
	SCOPE	37.53 ± 2.08	38.87 ± 2.69
	ETTRL Adv-Res	40.04 ± 0.55	41.60 ± 0.69
	TCCA (ours)	$\mathbf{41.10 \pm 0.91}$	$\mathbf{42.80 \pm 0.92}$
Llama3.2-1B-Instruct	Original TTRL	39.80 ± 3.04	41.73 ± 3.29
	TCCA (ours)	$\mathbf{42.82 \pm 0.26}$	$\mathbf{44.67 \pm 0.12}$

4.5 Credit construction and prototype count

The controlled removals in Table 3 examine the dependencies of the credit construction. RC-Off denotes TCCA without reasoning credit: it retains the target distribution and HOAS but removes reasoning credit; TCCA gains 1.33/1.40 sample/majority points, with positive differences in every pair. SoftVote-Off uses count voting and recomputes the selected target, answer statistics and references; TCCA gains 1.40/1.80 points. HOAS-Off removes only the prompt multiplier, lowering the means by 1.18/1.20 points. These comparisons connect the full method's gains to reasoning distinctions, target construction and prompt allocation. Together, these controlled comparisons quantify the gains of the complete credit construction over its individual removals. RC-Off and HOAS-Off remove downstream credit components while preserving the selected target, whereas SoftVote-Off changes the upstream target estimator and therefore recomputes both outcome statistics and target-conditioned references.

Table 3: **Component removals on Qwen2.5-0.5B/MATH-500.** Mean ± sample SD across Seed 1–3 under the primary protocol. Each removal retains the other components; deltas are TCCA minus the corresponding removal, in percentage points.

Configuration	Sample accuracy	Majority correctness
RC-Off	39.77 ± 0.15	41.40 ± 0.35
HOAS-Off	39.92 ± 0.30	41.60 ± 0.35
SoftVote-Off	39.70 ± 0.16	41.00 ± 0.53
TCCA (ours)	$\mathbf{41.10 \pm 0.91}$	$\mathbf{42.80 \pm 0.92}$
Δ vs. RC-Off	$+1.33$	$+1.40$
Δ vs. HOAS-Off	$+1.18$	$+1.20$
Δ vs. SoftVote-Off	$+1.40$	$+1.80$

A compact prototype set. TCCA with one prototype retains the reasoning-credit branch and refinement while allowing one prototype. In a dedicated step-150 comparison, we average two evaluation draws within each training run. Table 4 summarizes the dedicated prototype-count comparison. TCCA with at most two prototypes reaches 41.20/43.10% sample/majority correctness against 39.46/41.23% for TCCA with one prototype, corresponding to gains of 1.74/1.87 points. The improvement is positive in all three matched training runs; Appendix D reports the corresponding per-run endpoints.

4.6 Computational cost

On the same 8×A6000 setup, mean complete-run time increases from 21.38 hours for Original TTRL to 22.51 hours for TCCA, corresponding to a 1.053× ratio, or approximately 5.3% additional elapsed time. The auxiliary 0.6B encoder remains frozen. These measurements include initialization, training, in-training evaluation and checkpointing; Table 15 in Appendix E reports the per-run times and allocated GPU-hours.

Table 4: **Prototype-count comparisons on Qwen2.5-0.5B/MATH-500.** Paired Seed 1–3 means, averaging two evaluation draws within each run. The actual number of prototypes is selected up to the cap.

Paired configuration	Sample accuracy	Majority correctness
TCCA with one prototype	39.46	41.23
TCCA with at most two prototypes	41.20	43.10
Mean difference (points)	+1.74	+1.87

5 Design Analysis: Outcome and Reasoning Credit

We examine the information carried by reasoning credit and relate it to the prototype and prompt-allocation choices. A fixed pool of 64 responses for each MATH-500 input allows comparisons on identical text, with the first 32 retained as in training. These diagnostics apply the credit construction without policy updates. Appendix F gives the scoring protocol and complete results; Appendix C contains the single-run training analysis. Appendix A.5 derives the within-outcome distinctions and the normalization argument.

Two resolutions of credit. The two branches address different information losses in outcome-only TTRL. HOAS changes how strongly different prompts contribute after within-prompt outcome normalization, whereas reasoning credit changes how retained responses within the same prompt are distinguished. They therefore operate at two different resolutions—across prompts and within prompts—while remaining anchored to the same selected pseudo-target.

Reasoning credit adds distinctions unavailable to binary outcome coefficients. Among 282 retained groups containing both binary reward classes and nonzero reasoning-credit variation, 77.33% of the variance lies within those classes on average (Table 5). The 170 homogeneous groups consist of 164 all-$r_i = 1$ groups, all with nonzero reasoning credit, and six all-$r_i = 0$ groups, all with zero residual in this fixed pool. Because r_i is defined relative to the selected pseudo-target rather than gold correctness, this breakdown characterizes credit availability rather than correctness. Together, these fixed-pool results show that the reasoning branch can provide prompt-local distinctions that binary outcome coefficients do not express. The paired RC-Off comparison complements this structural evidence with a branch-level sample-accuracy gain of 1.33 points (39.77% to 41.10%; Table 3).

Table 5: **Credit variation on the fixed MATH-500 rollout pool.** Counts use 500 prompts and the 32 responses per prompt. The within-class fraction averages the ratio of within-class to total reasoning-credit variance over the 282 eligible mixed groups.

Quantity	Observation
Prompts with nonzero reasoning credit	446 / 500
Homogeneous groups with nonzero credit (all-$r_i = 1$ / all-$r_i = 0$)	164 / 164; 0 / 6
Mean within-outcome fraction of reasoning-credit variance	77.33%

Separate prototypes help when supported splits are present. In the single-run split/non-split analysis, using multiple rather than single prototypes changes sample accuracy by $+3.02$ points on split prompts and -0.63 on nonsplit prompts. The measured benefit is therefore concentrated where the reference set admits a supported geometric split. The split fraction rises from 3.40% for the base model to 43.00% for the final model. Appendix C.3 reports the full subgroup results, and Appendix G.2 shows how prototype use evolves over training.

Confidence provides a useful weighting signal. Figure 2 connects confidence to supervision quality in the single-run experiments. Rollout correctness generally increases across self-certainty deciles, while Soft Vote shows generally higher agreement between pseudo-label-derived and gold-answer-derived reward decisions than count voting over the observed training interval. Together, these observations support the use of confidence to weight the sampled evidence for target formation. Appendix C gives the measurement definitions.

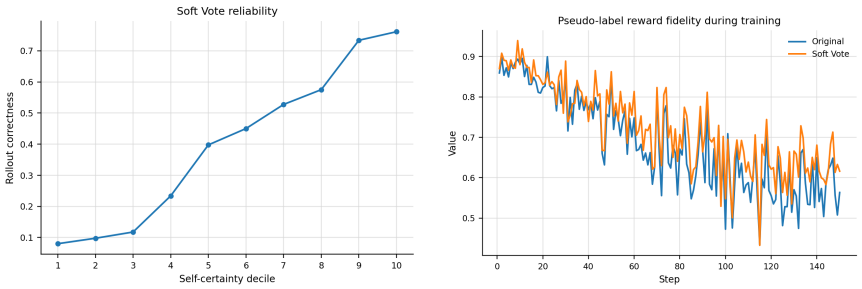


Figure 2: **Single-run evidence for confidence-weighted supervision.** Left: gold-answer correctness by self-certainty decile. Right: agreement between pseudo-label-derived and gold-answer-derived binary rewards during training. Seeds are not matched to Seed 1–3.

Prompt ambiguity directs outcome-update allocation. In the fixed low/medium/high-ambiguity partition, adding HOAS changes sample accuracy by -1.26, $+1.81$ and $+3.20$ points, with mean prompt weights 0.94, 1.01 and 1.06. The largest gain occurs in the high-ambiguity group, matching the intended weighting of outcome advantages within this fixed partition. These subgroup observations complement the paired HOAS-Off comparison, which measures the effect on overall learning. Table 6 shows the allocation and gains; Appendix C defines the fixed partition and mass ratio.

Table 6: **HOAS allocation under a fixed prompt partition.** Gain is the sample-accuracy difference in percentage points. Single runs without matched training seeds; weights and mass ratios describe allocation.

Ambiguity	Prompts	Gain	Mean weight	Scaled/base mass
Small	178	-1.26	0.94	1.04
Medium	162	$+1.81$	1.01	1.06
High	160	$+3.20$	1.06	1.08

6 Conclusion

TCCA turns a selected pseudo-target and its answer distribution into two complementary sources of learning credit. HOAS reallocates normalized outcome credit across prompts, while compact target-matching prototypes provide relative reasoning distinctions within each prompt. Combining the two at the advantage stage lets the policy use both answer agreement and rollout structure from the same sampled pool.

Across the broad single-run evaluations, TCCA yields higher point estimates than Original TTRL in all twelve settings. Matched three-seed MATH-500 experiments show positive TCCA-minus-Original differences in all three seeds on both Qwen2.5-0.5B and Llama3.2-1B-Instruct, with mean sample-accuracy gains of 2.14 and 3.02 points, respectively. On the Qwen setting, TCCA also exceeds the reproduced SCOPE and ETTRL Adv-Res means under the shared protocol, while component removals and prototype comparisons provide complementary support for the two credit pathways and compact reference design.

With a frozen encoder and approximately 5.3% additional complete-run time on the measured setup, TCCA provides a practical way to make self-generated supervision more informative for test-time learning. The reasoning branch remains a semantic comparison under self-generated supervision rather than a verifier of logical correctness, and the measured runtime overhead should not be interpreted as a universal cost ratio.

AI use statement

Generative AI tools were used to assist with manuscript language polishing, presentation refinement, code debugging, and the reproduction and verification of algorithmic implementations. They were

not used as a substitute for experimental execution or for making final scientific conclusions. All AI-assisted text, code, mathematical expressions, and technical interpretations were reviewed by the authors, and the reported experimental results were obtained from the authors' actual runs and analyses. The authors take full responsibility for the correctness, originality, and final content of this work.

Ethics statement

This work studies test-time reinforcement learning on publicly available reasoning benchmarks and does not involve human subjects, personally identifiable information, or private user data. The proposed method relies on self-generated supervision and may therefore inherit or reinforce errors, biases, or spurious correlations present in the underlying language model. In particular, pseudo-targets and semantic reasoning references should not be interpreted as guarantees of factual or logical correctness. We therefore recommend independent validation before applying such adaptation methods in high-stakes or safety-critical settings. The experiments also incur computational costs associated with rollout generation, model adaptation, and embedding inference.

Reproducibility statement

Sections 3–4 specify the method and experimental protocol. Appendices A–E provide the detailed algorithmic construction, shared training and evaluation settings, per-seed results, paired comparisons, and complete-run runtime measurements. Appendices F–G provide the fixed-pool diagnostics and prototype analyses used in the paper. We report the model backbones, benchmark schedules, rollout and retained-group sizes, optimizer settings, evaluation decoding parameters, training seeds, hardware configurations, and fixed algorithmic hyperparameters to facilitate reproduction.

References

Minghan Chen, Guikun Chen, Wenguan Wang, and Yi Yang. SEED-GRPO: Semantic Entropy Enhanced GRPO for Uncertainty-Aware Policy Optimization. *arXiv preprint arXiv:2505.12346*, 2025. URL https://arxiv.org/abs/2505.12346.

Ruiyi Ding, Yongxuan Lv, Xianhui Meng, Jiahe Song, Chao Wang, Chen Jiang, and Yuan Cheng. PRPO: Aligning Process Reward with Outcome Reward in Policy Optimization. *arXiv preprint arXiv:2601.07182*, 2026. URL https://arxiv.org/abs/2601.07182.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Yiran Guo, Lijie Xu, Jie Liu, Dan Ye, and Shuang Qiu. Segment Policy Optimization: Effective Segment-Level Credit Assignment in RL for Large Language Models. *arXiv preprint arXiv:2505.23564*, 2025. URL https://arxiv.org/abs/2505.23564.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Zhewei Kang, Xuandong Zhao, and Dawn Song. Scalable best-of-n selection for large language models via self-certainty. *Advances in neural information processing systems*, 38:19720–19745, 2025.

Jia Liu, ChangYi He, YingQiao Lin, MingMin Yang, FeiYang Shen, and ShaoGuo Liu. Ettrl: Balancing exploration and exploitation in llm test-time reinforcement learning via entropy mechanism. *arXiv preprint arXiv:2508.11356*, 2025.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, et al. Gdpo: Group reward-decoupled normalization policy optimization for multi-reward rl optimization. *arXiv preprint arXiv:2601.05242*, 2026.

Ambuj Mehrish and Sebastiano Vascon. Core: Consensus rewards via equilibrium for test-time reinforcement learning. *arXiv preprint arXiv:2608.09324*, 2026.

Manh Nguyen, Sunil Gupta, and Hung Le. Beyond majority voting: Efficient best-of-n with radial consensus score. *arXiv preprint arXiv:2604.12196*, 2026.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL `https://arxiv.org/abs/2412.15115`.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First conference on language modeling*, 2024.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Weiqin Wang, Yile Wang, Kehao Chen, and Hui Huang. Beyond majority voting: Towards fine-grained and more reliable reward signal for test-time reinforcement learning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 37251–37265, 2026.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Jingyu Xing, Chenwei Tang, Xinyu Liu, Deng Xiong, Shudong Huang, Wei Ju, Jiancheng Lv, and Ziyue Qiao. Rewarding the journey, not just the destination: A composite path and answer self-scoring reward mechanism for test-time reinforcement learning. *arXiv preprint arXiv:2510.17923*, 2025.

Dong Yan, Jian Liang, Yanbo Wang, Shuo Lu, Ran He, and Tieniu Tan. What if consensus lies? selective-complementary reinforcement learning at test time. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28957–28970, 2026.

Yongcan Yu, Lingxiao He, Jian Liang, Kuangpu Guo, Meng Wang, Qianlong Xie, Xingxing Wang, and Ran He. Understanding and mitigating spurious signal amplification in test-time reinforcement learning for math reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 37424–37436, 2026.

Zhaoning Yu, Will Su, Leitian Tao, Haozhu Wang, Aashu Singh, Hanchao Yu, Jianyu Wang, Hongyang Gao, Weizhe Yuan, Jason Weston, et al. Restrain: From spurious votes to signals–self-driven rl with self-penalization. *arXiv preprint arXiv:2510.02172*, 2025.

Nonghai Zhang, Weitao Ma, Zhanyu Ma, Jun Xu, Jiuchong Gao, Jinghua Hao, Renqing He, and Jingwen Xu. Silence the judge: Reinforcement learning with self-verifier via latent geometric clustering. *arXiv preprint arXiv:2601.08427*, 2026.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. Learning to reason without external rewards. In *International Conference on Learning Representations*, volume 2026, pp. 2548–2581, 2026.

Jiaru Zou, Ling Yang, Jingwen Gu, Jiahao Qiu, Ke Shen, Jingrui He, and Mengdi Wang. Reasonflux-prm: Trajectory-aware prms for long chain-of-thought reasoning in llms. *Advances in Neural Information Processing Systems*, 38:26764–26802, 2025.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Xinwei Long, Ermo Hua, Biqing Qi, et al. Ttrl: Test-time reinforcement learning. *Advances in Neural Information Processing Systems*, 38:131459–131483, 2025.

A Appendix: Additional Details of TCCA

This appendix provides additional details of the TCCA components introduced in Section 3. We explain how confidence supplies vote weights, how HOAS redistributes outcome credit, and how reference refinement and conservative clustering produce reasoning prototypes. The final subsections describe residual scaling and the reason for combining the two credit terms after normalization.

A.1 Self-Certainty and Soft Vote Weights

Following Kang et al. (2025) and Zhao et al. (2026), we use rollout-level self-certainty as a relative reliability signal. For response $y_i = (y_{i,1}, \ldots, y_{i,L_i})$, let $p_{i,t}(v) = \pi_\theta(v \mid x, y_{i,<t})$ denote the next-token distribution at position t over vocabulary $\mathcal{V}$, with $V = |\mathcal{V}|$, and let U_V denote the uniform distribution over $\mathcal{V}$. The rollout-level self-certainty is

$$s_i^{\mathrm{sc}} = \frac{1}{L_i} \sum_{t=1}^{L_i} D_{\mathrm{KL}}(U_V \| p_{i,t}) = -\frac{1}{L_i V} \sum_{t=1}^{L_i} \sum_{v \in \mathcal{V}} \log(V p_{i,t}(v)) \,. \tag{5}$$

Self-certainty is not used as a standalone reward. Soft Vote converts these scores into relative vote weights $\omega_i = s_i^{\mathrm{sc}} / \sum_{j=1}^{N_{\mathrm{vote}}} s_j^{\mathrm{sc}}$, with uniform weights if the denominator is numerically zero. The resulting ω_i are used in Eq. 1 to construct $q_x(a)$ and select the pseudo-target $a_x^\star$.

A.2 Normalization

GRPO normalizes binary outcome rewards within the retained training group T_x to obtain A_i^{base}. Responses with equal rewards in the same group therefore receive equal base advantages.

HOAS uses the positive-mass support $\mathcal{A}_x = \{a : q_x(a) > 0\}$ and $K_x = \max(2, |\mathcal{A}_x|)$:

$$d_x = -\frac{\sum_{a \in \mathcal{A}_x} q_x(a) \log q_x(a)}{\log K_x}. \tag{6}$$

A single supported label gives zero entropy. Labels with zero vote mass are not counted in K_x; in particular, unobserved answer options are not added to the support solely because the task is multiple-choice.

The batch mean is $\bar{d} = |B|^{-1} \sum_{x \in B} d_x$. Eq. 2 first clips the linear adjustment to $[0.8, 1.2]$ and then normalizes the resulting weights to batch mean one. The clipping bounds apply to $\widetilde{w}_x$, not necessarily to the final w_x. Mean normalization controls the average multiplier; it does not guarantee an unchanged total advantage magnitude or gradient norm. HOAS multiplies the normalized outcome advantage, not the raw reward.

A.3 Conservative Multi-Centroid Construction

The reference representation balances two aims: retaining separated regions of reasoning traces and avoiding weakly supported fragments. A second center is used only when the target-matching traces support a separated group; otherwise, one refined centroid summarizes the set.

Reference and scoring views. For target-matching responses, deterministic reference cleaning retains the main reasoning content while removing terminal answer spans and lightweight non-reasoning artifacts, such as prompt echoes or degenerate repetitions. The scoring view is cleaned

less aggressively. This asymmetric construction reduces artifacts in prototype estimation without removing all corresponding distinctions from response scoring.

The frozen encoder Enc produces L2-normalized reference embeddings e_i^{ref} and scoring embeddings e_i^{roll}. Reference embeddings are constructed from target-matching traces; scoring embeddings are constructed for all scoreable retained responses.

Reference-set weights. Let $\widetilde{P}_x \subseteq P_x$ contain the reference indices remaining after trace-level preprocessing, before robust refinement. Define $s_{\min,x} = \min_{j\in\widetilde{P}_x} s_j^{\mathrm{sc}}$ and $s_{\max,x} = \max_{j\in\widetilde{P}_x} s_j^{\mathrm{sc}}$. When $s_{\max,x} > s_{\min,x}$, use

$$\widetilde{s}_i^{\mathrm{sc}} = \frac{s_i^{\mathrm{sc}} - s_{\min,x}}{s_{\max,x} - s_{\min,x}}, \qquad \nu_i = \frac{\widetilde{s}_i^{\mathrm{sc}}}{\sum_{j\in\widetilde{P}_x} \widetilde{s}_j^{\mathrm{sc}}}, \quad i \in \widetilde{P}_x. \tag{7}$$

When all reference self-certainty scores are equal, use uniform reference weights; their confidence-distance contribution is zero. The weights ν_i are used for reference refinement, split-support estimation and prototype construction. They are distinct from the full-pool voting weights ω_i.

Robust reference refinement. For a reference group with index set S, form a provisional confidence-weighted unit center from $\{\nu_i e_i^{\mathrm{ref}} : i \in S\}$. Let h_i be the cosine similarity of reference i to this center, and let $\mathcal{H}_S = (h_i)_{i\in S}$ denote the indexed collection of similarities. Define

$$m_S = \mathrm{median}(\mathcal{H}_S), \qquad \mathrm{MAD}(\mathcal{H}_S) = \mathrm{median}\big((|h_i - m_S|)_{i\in S}\big). \tag{8}$$

Retain references satisfying $h_i \geq m_S - 1.4826\,\mathrm{MAD}(\mathcal{H}_S)$. The center is confidence-weighted, whereas the median and MAD in this refinement rule are ordinary sample statistics. The same refinement is applied once before split selection and once within each accepted group before prototype estimation.

Conservative clustering and supported splits. Average-linkage hierarchical clustering uses

$$D(i,j) = \alpha\big(1 - \cos(e_i^{\mathrm{ref}}, e_j^{\mathrm{ref}})\big) + 2\,(1-\alpha)\,\big|\widetilde{s}_i^{\mathrm{sc}} - \widetilde{s}_j^{\mathrm{sc}}\big|, \qquad \alpha = 0.9. \tag{9}$$

The factor 2 matches the range of the confidence-distance term to that of cosine distance: since $\widetilde{s}_i^{\mathrm{sc}} \in [0,1]$, $2|\widetilde{s}_i^{\mathrm{sc}} - \widetilde{s}_j^{\mathrm{sc}}| \in [0,2]$, matching the range of $1 - \cos(e_i^{\mathrm{ref}}, e_j^{\mathrm{ref}})$. The semantic term provides the main distance, while the confidence term discourages grouping traces with markedly different reliability scores.

We inspect dominant candidate groups in the upper-level clustering hierarchy. Each candidate group must contain at least three references and at least 0.15 of the reference mass. For a candidate cluster C, define its reference mass as $m(C) = \sum_{i\in C} \nu_i$. Each accepted candidate must satisfy $|C| \geq 3$ and $m(C) \geq 0.15$.

For two candidate groups, let b be their mean cross-group pairwise distance, and let $\mathcal{W}$ collect the within-group pairwise distances, excluding self-pairs. The separation requirement is

$$b \geq \mathrm{median}(\mathcal{W}) + 3 \cdot 1.4826\ \mathrm{MAD}(\mathcal{W}). \tag{10}$$

The current two-prototype configuration retains a supported split when these conditions are met; otherwise it uses one refined group.

Prototype construction. Let $C_{x,m}$ contain the reference indices in accepted group m after its within-group refinement. Its unit prototype is

$$c_{x,m} = \frac{\sum_{j\in C_{x,m}} \nu_j e_j^{\mathrm{ref}}}{\left\|\sum_{j\in C_{x,m}} \nu_j e_j^{\mathrm{ref}}\right\|_2}, \qquad m = 1, \ldots, M_x. \tag{11}$$

The weights are the reference weights defined in Eq. 7. When usable reference directions can be constructed, $M_x \in \{1, 2\}$ in the full configuration. Eq. 3 compares each scoreable response with its nearest retained prototype. Unusable references supply no reasoning residual.

Table 7: **Principal fixed settings for TCCA.** Settings are fixed except for the explicitly varied components in ablations.

Quantity	Setting
HOAS coefficient / raw-weight clamp	$\beta = 0.5$ / $[0.8, 1.2]$
Prototype cap	2
Semantic-distance coefficient	$\alpha = 0.9$
Minimum candidate size / mass fraction	3 / 0.15
Refinement MAD multiplier / scale	1 / 1.4826
Split-separation MAD multiplier / scale	3 / 1.4826
Reasoning coefficient / clip / stabilizer	0.05 / 2 / 10^{-6}

A.4 Prompt-Relative Reasoning Credit

Let $F_x = \{i \in T_x : s_i^{\mathrm{rc}} \text{ is finite}\}$ be the scoreable retained indices. Let μ_x^{rc} and σ_x^{rc} denote the mean and population standard deviation of their similarity scores. For $i \in F_x$, define

$$\Delta_i^{\mathrm{rc}} = \lambda_{\mathrm{rc}} \operatorname{clip}\left(\frac{s_i^{\mathrm{rc}} - \mu_x^{\mathrm{rc}}}{\sigma_x^{\mathrm{rc}} + \epsilon_{\mathrm{rc}}}, -c_{\mathrm{rc}}, c_{\mathrm{rc}} \right). \tag{12}$$

We use $\lambda_{\mathrm{rc}} = 0.05$ and $c_{\mathrm{rc}} = 2$; $\epsilon_{\mathrm{rc}} > 0$ is a denominator stabilizer. Unscoreable responses and prompts without usable references receive zero reasoning residual.

Normalization is performed across finite scores in the retained group, not across the full voting pool or separately within binary reward classes. Identical finite scores give zero residual. Clipping bounds each contribution by $|\Delta_i^{\mathrm{rc}}| \le 0.1$. The pre-clipping standardized scores are zero-mean over the finite retained scores, but clipping does not in general preserve a zero retained-group mean. Appendix F quantifies this post-clipping mean shift on the fixed rollout pool. The residual is added as in Eq. 4.

A.5 Why combine credit after group normalization?

For two retained responses $i, j \in T_x$ from the same prompt, equal binary rewards give equal base advantages. Their combined coefficients can nevertheless differ:

$$A_i^{\mathrm{final}} - A_j^{\mathrm{final}} = \Delta_i^{\mathrm{rc}} - \Delta_j^{\mathrm{rc}} \quad \text{if } i, j \in T_x \text{ and } r_i = r_j. \tag{13}$$

Thus a homogeneous outcome group can retain reasoning credit when usable references yield nonconstant similarity scores. For prompt allocation, a common positive reward multiplier cancels under exact standardization:

$$\frac{w_x r_i - w_x \bar{r}_x}{w_x \sigma_x^r} = \frac{r_i - \bar{r}_x}{\sigma_x^r}, \qquad w_x > 0, \quad \sigma_x^r > 0, \tag{14}$$

where $\bar{r}_x$ and σ_x^r are the group's reward mean and standard deviation. HOAS therefore weights the advantage after normalization. These statements concern outcome and reasoning credit; the separately added KL loss does not enter this normalization and can still contribute a gradient when outcome advantages vanish. This cancellation is exact for nondegenerate standardization without an additive numerical stabilizer. It explains the placement of HOAS after normalization; it is not a claim that all stabilized implementations are exactly scale-invariant.

B Appendix: Shared Settings and Single-Run Experiments

The broad experiments use single runs without matched training seeds to examine benchmark coverage and the credit construction. Paired Seed 1–3 experiments then test repeatability under the shared training recipe.

B.1 Common training and evaluation settings

We inherit the TTRL training hyperparameters and evaluation recipe (Zuo et al., 2025). Each benchmark supplies unlabeled adaptation inputs, and the same inputs are used for evaluation with

gold answers. Table 8 collects the common settings rather than repeating them for single-run and paired experiments. The paired MATH-500 runs use the fixed step-150 endpoint. Small-model experiments use 8×A6000, and the 7B experiments use 8×A800.

Table 8: **Shared TTRL training and evaluation recipe.** The common settings apply to both the single-run and paired experiments; benchmark schedules are listed together below.

Setting	Value
Optimizer / peak learning rate	AdamW / 5×10^{-7}
Learning-rate schedule	Cosine
Vote pool / retained training group	64 / first 32 responses
Policy KL loss coefficient	0.001
Maximum response length	3072 tokens
Evaluation samples / temperature / top-p	16 / 0.6 / 0.95
Primary metric	Sample accuracy (sampled pass@1)
Paired MATH-500 endpoint / prompt batch	150 optimizer steps / 32
Benchmark	Single-run epochs / prompt batch
MATH-500 (Hendrycks et al., 2021)	10 / 32
AIME24 (TTRL suite) (Zuo et al., 2025)	80 / 8
AMC (TTRL suite) (Zuo et al., 2025)	30 / 8; Llama3.2-1B: 15 / 8
GPQA-Diamond (Rein et al., 2024)	8 / 16

B.2 Cumulative credit construction

Table 9 presents the single-run configuration path. Reasoning credit uses one refined prototype until the multi-prototype component is enabled. These exploratory results are separate from the paired component removals in Table 3.

Table 9: **Cumulative configuration path on Qwen2.5-0.5B/MATH-500.** Pass@1 (%); single runs without matched training seeds. TTRL denotes the outcome-only baseline (Zuo et al., 2025). A check mark denotes an enabled component; blank cells denote absence.

Base	TTRL	HOAS	Soft Vote	Reasoning credit	Multi-prototype	MATH-500
✓						13.94
✓	✓					36.81
✓	✓	✓				37.96
✓	✓	✓	✓			40.71
✓	✓	✓	✓	✓		41.06
✓	✓	✓	✓	✓	✓	**42.00**

B.3 Prototype-cap comparison

Table 10 includes the base and outcome-only reference rows alongside the three prototype caps. The two-center configuration has the highest score in this exploration. The dedicated paired comparison then evaluates one versus two centers, with the actual prototype count selected adaptively up to the cap.

Allowing a third prototype reduces pass@1 from 42.00% to 39.23%, consistent with over-fragmentation of the relatively small target-matching reference sets. Additional directions may be supported by fewer traces and become less stable. We therefore use a conservative cap of two and validate the one-versus-two choice separately with paired runs.

C Appendix: Analyses of the Single-Run Experiments

We examine how the selected target organizes supervision, prompt allocation and reasoning references. The analyses below use single-run measurements without matched training seeds. The paired removals provide a separate test of repeatability.

Table 10: **Prototype-cap exploration.** Qwen2.5-0.5B/MATH-500 pass@1 (%), without the matched-seed protocol.

Configuration	MATH-500
Base model	13.94
Original TTRL (Zuo et al., 2025)	36.81
TCCA with one prototype	41.06
TCCA with at most two prototypes	42.00
TCCA with at most three prototypes	39.23

C.1 Confidence as a relative weighting signal

Figure 2 relates rollout confidence to correctness and to the rewards constructed from the selected answer. In the left panel, rollouts are sorted by self-certainty and divided into ten equal-frequency bins. Correctness generally rises with confidence. This supports confidence as a relative weighting signal in this sample, without establishing that its magnitude is a calibrated probability of correctness.

The right panel measures *reward accuracy*: the fraction of rollout-level binary reward decisions that agree when using the pseudo-target versus the gold answer. The Soft Vote curve is generally higher than the count-vote curve over the observed training interval. This is reward-decision agreement, which can include jointly negative decisions; it should not be read as direct pseudo-target accuracy. Together, the panels support the use of relative confidence in constructing supervision from the rollout pool.

C.2 Answer ambiguity and prompt-level allocation

The analysis fixes a prompt partition using the Soft Vote model and reuses it for Soft Vote and Soft Vote + HOAS. Table 6 shows increasing mean weights across ambiguity groups. The advantage-mass ratio sums absolute advantages after scaling and divides by the corresponding sum before scaling within each group. Its trend also increases, although it need not equal the unweighted mean prompt multiplier.

The high-ambiguity group improves most, while the small-ambiguity group declines. This pattern is consistent with the intended redistribution of outcome credit, but it does not show that entropy always identifies useful difficulty: disagreement can also reflect unreliable targets.

C.3 Where multiple prototypes help

This analysis asks *where* a second prototype helps, complementing the overall prototype-cap comparison. In a synchronized final-step evaluation slice, prompts are partitioned by whether their reference set admits a supported geometric split. Single- and multi-centroid predictions are evaluated against gold answers within the same prompt groups. Table 11 reports the subgroup results.

Table 11: **Reasoning-reference analysis.** Sample accuracy (%) in the split/non-split analysis slice, without the matched-seed protocol. Deltas are multi- minus single-centroid.

Prompt group	Single centroid	Multiple centroids	Difference
Split	48.98	52.00	$+3.02$
Nonsplit	35.09	34.46	-0.63

Multiple prototypes gain 3.02 points on split prompts and lose 0.63 points on nonsplit prompts. The measured benefit is therefore concentrated where the reference set admits a supported geometric split. These are subgroup scores from the analysis slice, not overall benchmark scores. The observed split fraction rises from 3.40% for the base model to 43.00% for the final model.

These observations are consistent with preserving separated reference regions, but a geometric split alone does not identify logically valid reasoning families. Increased split frequency could also reflect changes in support size or representation. Appendix G.2 reports how prototype use changes over

training. The matched prototype comparison in Section 4.5 provides the separate performance test of allowing a second center.

D Appendix: Experimental Settings and Per-Run Results

All repeated experiments use the TTRL training and evaluation recipe summarized in Table 8. Results are evaluated at the fixed step-150 endpoint.

Aggregation. For each backbone, the mean and sample standard deviation are computed across three matched training seeds. Table 12 reports the per-seed endpoints for Original TTRL and TCCA on the two repeated MATH-500 settings underlying the main paired comparison. For Llama3.2-1B-Instruct, the TCCA-minus-Original paired differences for Seeds 1–3 are $+6.13/+6.60$, $+0.64/+0.60$, and $+2.29/+1.60$ points in sample accuracy / majority correctness, respectively. Table 13 reports the corresponding paired differences for the Qwen2.5-0.5B baselines and component removals. The separate prototype comparison averages two decoding draws within each training run and is reported in Table 14.

Table 12: **Per-seed step-150 endpoints for repeated MATH-500 evaluations.** Each seed cell reports sample accuracy / majority correctness (%). The three matched runs underlie the aggregated results in the main text.

Backbone	Method	Seed 1	Seed 2	Seed 3
Qwen2.5-0.5B	Original TTRL	39.39 / 41.00	39.33 / 40.60	38.16 / 39.60
	TCCA	41.29 / 43.00	41.90 / 43.60	40.11 / 41.80
Llama3.2-1B-Instruct	Original TTRL	36.40 / 38.00	42.25 / 44.20	40.75 / 43.00
	TCCA	42.53 / 44.60	42.89 / 44.80	43.04 / 44.60

Component removals. RC-Off removes reasoning credit while retaining Soft Vote and HOAS. HOAS-Off removes the prompt multiplier. SoftVote-Off uses count voting and recomputes answer statistics and target-matching references. TCCA with one prototype retains the reasoning-credit branch but restricts the representation to one prototype. All remaining settings follow the corresponding full configuration.

Baseline implementation. We follow the original SCOPE target and reward construction (Wang et al., 2026) and the original EAR formulation for ETTRL Adv-Res (Liu et al., 2025). SCOPE computes targets and rewards from all 64 responses; our common trainer retains the first 32 for optimization, whereas the original SCOPE pipeline trains on all 64. EAR is applied after base GRPO advantage construction in the common parallel-rollout trainer.

Table 13: **Per-seed paired differences on Qwen2.5-0.5B/MATH-500.** TCCA minus comparator, in percentage points; positive values favor TCCA. S = sample accuracy; M = majority correctness.

	Seed 1		Seed 2		Seed 3	
Comparator	S	M	S	M	S	M
Original	$+1.90$	$+2.00$	$+2.58$	$+3.00$	$+1.95$	$+2.20$
SCOPE (Wang et al., 2026)	$+5.96$	$+7.20$	$+4.09$	$+3.60$	$+0.65$	$+1.00$
ETTRL Adv-Res (Liu et al., 2025)	$+1.85$	$+2.20$	$+1.38$	$+1.60$	-0.04	-0.20
RC-Off	$+1.36$	$+1.40$	$+2.26$	$+2.00$	$+0.38$	$+0.80$
HOAS-Off	$+1.71$	$+1.80$	$+1.75$	$+1.80$	$+0.08$	$+0.00$
SoftVote-Off	$+1.49$	$+1.40$	$+2.11$	$+2.80$	$+0.59$	$+1.20$

Prototype comparison. The dedicated comparison uses existing step-150 checkpoints, with 16 samples per prompt in each of two decoding draws. The per-run endpoints in Table 14 average the two draws and underlie the comparison in Section 4.5.

Table 14: **Per-run endpoints for the separate two-draw comparison of TCCA prototype caps.**

Run	One prototype sample / majority	At most two prototypes sample / majority	Difference
Seed 1	39.21 / 41.10	41.36 / 43.40	+2.16 / +2.30
Seed 2	39.64 / 41.30	41.76 / 44.00	+2.12 / +2.70
Seed 3	39.55 / 41.30	40.48 / 41.90	+0.93 / +0.60

E Appendix: Complete-Run Runtime

Table 15: Elapsed runtime on the same 8×A6000 configuration.

Run	Original hours	TCCA hours	Ratio
Seed 1	21.34	22.53	1.056
Seed 2	21.42	22.54	1.052
Seed 3	21.38	22.46	1.051
Mean	21.38	22.51	1.053
Allocated GPU-hours (mean)	171.03	180.06	1.053

The ratio divides unrounded mean runtimes. Allocated GPU-hours equal eight times elapsed hours. The measured interval includes initialization, training (including rollout generation, reference construction and policy updates), in-training evaluation and checkpointing. It excludes queue waiting and separate post-training evaluation.

F Appendix: Fixed-Pool Diagnostics of Reasoning Credit

These analyses reuse fixed responses or perform forward-only rescoring. They involve zero optimizer updates. The historical pool contains 500 MATH-500 prompts and 64 responses per prompt from a fixed existing Qwen2.5-0.5B policy. The pool's native generation confidence was not saved. Confidence values for this offline analysis were recomputed by teacher forcing on the saved token sequences. These recomputed values should be distinguished from confidence recorded during the original rollout generation. The credit diagnostics use fixed text and representations.

Within-outcome variation. For prompt x, let r_i be the binary pseudo-reward on retained responses and $\bar{\Delta}_{x,r_i}$ the reasoning-credit mean within its reward class. Let $\bar{\Delta}_x$ denote the mean over the retained group T_x. We compute

$$V_x = \frac{\sum_{i \in T_x} (\Delta_i^{\mathrm{rc}} - \bar{\Delta}_{x,r_i})^2}{\sum_{i \in T_x} (\Delta_i^{\mathrm{rc}} - \bar{\Delta}_x)^2}. \tag{15}$$

Only groups with both reward classes and nonzero total reasoning-credit variation enter this ratio: 282 groups have mean $V_x = 77.33\%$, median 84.64%, and pooled within/total sums of squares 78.69%. For a homogeneous group with nonzero reasoning-credit variation, the within-class ratio is identically $V_x = 1$, so homogeneous groups are reported separately rather than included in this aggregate. Among the 170 homogeneous groups, all 164 all-$r_i = 1$ groups contain nonzero reasoning credit, whereas the six all-$r_i = 0$ groups have zero residual in this fixed pool. Because r_i is defined relative to the pseudo-target rather than the gold answer, this split is a diagnostic of credit availability, not of reasoning correctness. The RC-Off training comparison evaluates the reasoning branch as a whole and does not isolate the causal contribution of homogeneous groups alone.

Post-clipping mean shift. Because the standardized reasoning scores are clipped without a subsequent re-centering step, the executed residual need not have exactly zero mean within a retained group. For prompt x, define

$$b_x = \frac{1}{|T_x|} \sum_{i \in T_x} \Delta_i^{\mathrm{rc}}, \tag{16}$$

Table 16: **Different derivations can share one outcome target.** Worked algebraic illustrations of alternative reasoning routes.

Example 1. For $|x-4|-10=2$, find the product of the roots. **Answer:** -128.

Case split	**Quadratic rewrite**
$\lvert x-4\rvert = 12$	$(x-4)^2 = 144$
$x-4 = \pm 12$	$x^2 - 8x - 128 = 0$
$x \in \{16, -8\}$	$(x-16)(x+8) = 0$

Example 2. Solve $\frac{2}{3} = \frac{4}{x-5}$ for $x \neq 5$. **Answer:** 11.

Clear denominators	**Retain fractional coefficients**
$2(x-5) = 3 \cdot 4$	$\frac{2}{3}(x-5) = 4$
$2x - 10 = 12$	$\frac{2}{3}x - \frac{10}{3} = 4$
$x = 11$	$x = 11$

where zero residuals are included.

Across the 500-prompt fixed pool, the mean signed shift is 0.00216 and the mean absolute shift is 0.00217. The median, 95th percentile, and maximum of $|b_x|$ are 0.00205, 0.00521, and 0.00557, respectively. The mean fraction of retained responses reaching the clipping boundary is 4.91%.

Among the 446 groups with nonzero reasoning credit, the ratio

$$\frac{|b_x|}{|T_x|^{-1}\sum_{i\in T_x}|\Delta_i^{\mathrm{rc}}|}$$

has median 6.68% and 95th percentile 28.44%. These statistics characterize the residual space only; they do not measure effects on gradient norms, KL behavior, or final task performance.

G Appendix: Additional Prototype Illustrations and Diagnostics

The single-run experiments illustrate how prototypes represent and use target-supporting trajectories during adaptation. We first show two worked derivations, then examine reference selection and prototype use. The paired comparison in Table 14 separately tests the performance of retaining a second center.

G.1 A shared answer can support different derivations

Table 16 shows two ways intermediate expressions can differ while the final answer remains fixed. For the first problem, a case split and a quadratic rewrite recover the same roots. For the second, clearing denominators immediately and retaining fractional coefficients both yield $x = 11$. These worked illustrations motivate a reasoning representation that can retain alternatives around one outcome target.

The reasoning-credit construction addresses this representation choice through nearest-prototype scoring. When the target-matching embeddings support two separated regions, a response can be close to either prototype. A single average center compresses these regions into one direction. The worked examples explain the design motivation; the measured advantage of the two-center configuration is given in Table 14.

G.2 Prototype use adapts to the reference support

To complement the final-step split/non-split analysis in Appendix C.3, Figure 3 tracks how the supported prototype structure changes during training.

The second prototype is used selectively throughout the observed training trajectory: the split rate ranges from 12.50% to 78.13% across updates. The dominant-cluster fraction generally falls from its

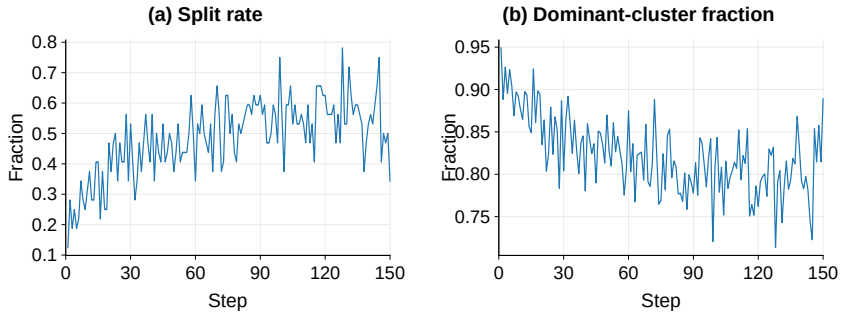


Figure 3: **Prototype use varies with the sampled reference support.** Split rate and dominant-cluster fraction over steps 1–150 of one Qwen2.5-0.5B/MATH-500 full-method run.

initial high concentration while remaining uneven, reflecting changing reference support. Thus, the two-center option does not require every prompt to split into balanced groups.