ONE OBJECTIVE, TWO REGIMES: UNIFYING LANGUAGE MODEL PRE-TRAINING AND REINFORCEMENT POST-TRAINING WITH LEARNED TOKEN CREDIT

Anonymous authors
Paper under double-blind review

ABSTRACT

Pre-training and reinforcement post-training are usually treated as different algorithms. We show that both optimize one weighted autoregressive objective, differing only in the trajectory distribution and per-token learning value. This view motivates UNICREDIT, a lightweight causal credit decoder inside the policy. From detached final hidden states, it learns reward, value, advantage, and importance targets; rule functions and held-out semantic chunks provide explicit supervision. No policy-sized reward model or critic is required. On matched FineWeb-Edu runs, UNICREDIT improves the six-task zero-shot mean from 42.80 to 43.58 at 455M parameters and from 52.54 to 53.23 at 1.5B. We further reproduce a Qwen2.5-Math-7B DAPO baseline reaching 36.0% AIME-2024 avg@32 and study a preliminary UNICREDIT trajectory reaching 37.8%. Diagnostics show why calibration matters: importance validity reaches roughly 91%, while advantage and verbal self-score correlations remain modest. The framework therefore unifies generation and judgment in one policy, while admitting learned credit only after its predictive quality is demonstrated.

1 INTRODUCTION

Modern language-model development is organized as a sequence of stages. Pre-training fits next-token likelihood on a large, mostly fixed corpus; supervised fine-tuning changes the data distribution; preference or reinforcement learning (RL) then samples from the model and optimizes a reward-weighted surrogate. This decomposition has been operationally successful (Brown et al., 2020; Ouyang et al., 2022; Qwen Team, 2024; DeepSeek-AI, 2025), but it obscures a common computational primitive: every stage ultimately changes the probability of observed tokens by multiplying their score-function gradient by a scalar learning signal.

We make this common structure explicit. Let a trajectory $\tau = (a_1, \ldots, a_T)$ be drawn from an arbitrary training distribution q, and let ω_t denote the value of learning token a_t in state $s_t = a_{<t}$. We study

$$\mathcal{L}_{\mathrm{UL}}(\theta; q, \omega) = -\mathbb{E}_{\tau \sim q}\left[\frac{1}{Z(\tau)} \sum_{t=1}^{T} \omega_t(\tau) \log \pi_\theta(a_t \mid s_t)\right], \tag{1}$$

where Z specifies token- or sequence-level normalization. Standard pre-training sets $q = \mathcal{P}_{\mathrm{data}}$ and $\omega_t = 1$. Policy gradient draws trajectories from a behavior policy and sets ω_t to a stopped-gradient product of an importance ratio and an advantage. PPO, RLOO, GRPO, DAPO, Dr. GRPO, and GSPO further change the baseline, clipping active set, normalization, or sampling distribution (Schulman et al., 2017; Ahmadian et al., 2024; Shao et al., 2024; Yu et al., 2025; Liu et al., 2025b; Zheng et al., 2025). Thus the apparent CE–RL divide can be represented by two questions: *what data should be learned* (q), and *how much should each token be learned* (ω).

This mathematical reduction is useful only if it leads to a concrete learner. Our implementation, UNICREDIT, estimates token credit directly from the policy's final hidden states using a small causal Transformer. The module predicts token reward, value, advantage, and importance ratio. It shares the representation learned by the policy but not the auxiliary gradient: h_t is unconditionally detached before entering the credit decoder. During pre-training, targets are built from multi-scale

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.

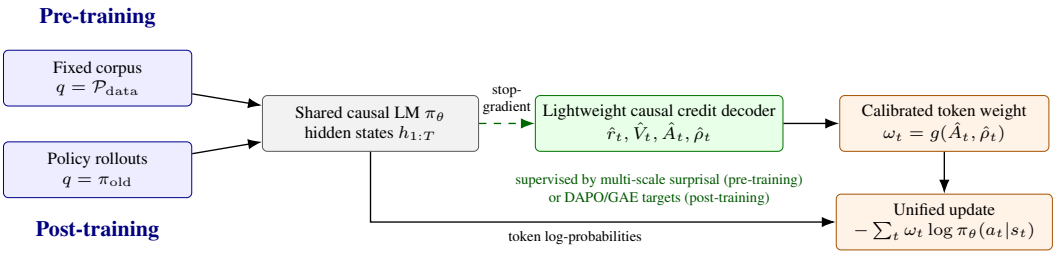

Figure 1: **One optimization interface for two regimes.** Fixed text and policy rollouts differ in q, while their learning algorithms differ in the construction of ω_t. A detached causal decoder learns token credit from the shared policy representation. Only calibrated, guard-approved predictions affect the policy objective. Dashed arrows denote an enforced stop-gradient boundary.

normalizations of gold-token log likelihood. During post-training, the targets come from quantities already available in DAPO—rule reward, group-relative advantage, old log probabilities, and a masked generalized advantage estimator (GAE). No separate reward model or full critic is instantiated.

The separation between *learning a credit estimator* and *trusting it for policy optimization* is central. A low regression loss can be misleading when most targets are near zero, and a per-sequence signal can be destroyed by global whitening. We therefore preserve each response's group-level outcome in the token targets, measure correlation and accuracy on nontrivial targets, normalize importance ratios to mean one within a response, and admit the learned branch only when quality and PPO stability guards pass. A failed guard reduces exactly to the baseline objective.

We evaluate both regimes rather than presenting the unification only as notation. First, we run matched CE-only and UNICREDIT pre-training at 455M/20B tokens and 1.5B/50B tokens on FineWeb-Edu (Penedo et al., 2024). Second, we integrate the same principle into a reproduced Qwen2.5-Math-7B DAPO run on 17,917 math prompts. We report both positive and negative evidence: learned credit improves the matched language-model mean, while post-training diagnostics show that accurate-looking marginal statistics do not imply safe policy credit. In particular, an intentionally aggressive 80% learned branch collapses, while our detached and readiness-gated path leaves the original DAPO gradient unchanged until the head is genuinely useful.

Our contributions are:

- a unified (q, ω) formulation that recovers CE, policy gradient, PPO, GRPO, and DAPO at the gradient level;
- a parameter-efficient causal credit decoder with explicit pre-training and post-training targets, eliminating the need for a separate full-size critic in the studied setting;
- stability conditions—gradient isolation, per-sequence outcome preservation, nontrivial-target readiness, and trust-region guards—derived from observed collapse modes; and
- end-to-end evidence from 20.97B-token pre-training and 7B DAPO, including all negative ablations needed to delimit the method's current claims.

2 A UNIFIED AUTOREGRESSIVE VIEW

2.1 TWO AXES: DATA DISTRIBUTION AND LEARNING VALUE

Equation 1 separates an algorithm into q and ω. The former can be a static corpus, a curated subset, an on-policy rollout distribution, or a replay mixture. The latter can be uniform, quality-weighted, or signed. This is more than a visual analogy: with stop-gradient on ω, the update always has the form

$$-\nabla_\theta \mathcal{L}_{\text{UL}} = \mathbb{E}_{\tau \sim q}\left[\frac{1}{Z}\sum_t \omega_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right]. \tag{2}$$

The sign and magnitude of ω_t determine whether an observed token is reinforced or suppressed; q determines which states and actions are available for learning.

Table 1: Existing training algorithms as choices of the data distribution q, token weight ω, and normalization Z.

Method	$q(\tau)$	ω_t (stop-gradient)	Z
Pre-training	corpus	1	valid tokens
Weighted pre-training	corpus	positive calibrated credit	valid tokens
Policy gradient	behavior policy	$\rho_t A_t$	trajectories
PPO	behavior policy	$g_t^{\text{clip}} \rho_t A_t$	valid actions
GRPO	grouped rollouts	$g_t^{\text{clip}} \rho_t A_i^G$	responses
DAPO	dynamic rollouts	asymmetric-clip $\rho_t A_i^G$	response tokens
UNICREDIT	corpus or rollouts	guarded $g(\hat{A}_t, \hat{\rho}_t)$	per sequence

Pre-training. For $q = \mathcal{P}_{\text{data}}$, $Z = T$, and $\omega_t = 1$, Equation 1 is exactly mean token cross-entropy. Non-uniform positive weights recover data reweighting without reversing gold-token gradients. This restriction matters: a signed advantage copied directly from RL would lower the likelihood of valid corpus tokens and is not generally appropriate for ordinary pre-training.

Policy gradient and PPO. Let $q = \mu$ be a behavior policy, $\rho_t = \pi_\theta(a_t \mid s_t)/\mu(a_t \mid s_t)$, and A_t an advantage. Setting $\omega_t = \text{sg}[\rho_t A_t]$ makes Equation 2 the policy-gradient estimator. For PPO, define $g_t^{\text{clip}} \in \{0, 1\}$ to indicate that the unclipped term is active for the sign of A_t. Then $\omega_t = \text{sg}[g_t^{\text{clip}} \rho_t A_t]$ produces the same local gradient as the clipped surrogate on a fixed rollout batch (Schulman et al., 2017). The objectives need not have identical scalar values; their policy gradients agree away from nondifferentiable clipping boundaries.

GRPO and DAPO. GRPO replaces an explicit critic by a standardized reward within a group of responses, $A_i^G = (R_i - \bar{R}_G)/(\sigma_G + \epsilon)$ (Shao et al., 2024). DAPO adds asymmetric Clip-Higher, dynamic sampling, token-level aggregation, and overlong shaping (Yu et al., 2025). In our coordinates, group-relative reward changes ω, dynamic sampling changes q, and token-level aggregation changes Z. Table 1 summarizes the mapping.

2.2 WHY LEARN TOKEN CREDIT?

Terminal reward broadcasts one scalar over a long response. It cannot distinguish a decisive derivation from boilerplate, correction, or an accidental correct answer. A full critic can supply dense credit but duplicates billions of parameters. Process reward models provide local supervision but require step annotations or separately generated labels (Lightman et al., 2024). We instead ask whether the policy representation already contains enough information for a small causal decoder to estimate the training quantities produced by existing algorithms. This is intentionally a modest claim: the decoder compresses available supervision; it does not create correctness information from nothing.

3 METHOD

3.1 DETACHED CAUSAL CREDIT DECODER

Let $h_{1:T} \in \mathbb{R}^{T \times d}$ be final-layer policy states. We project detached states to $d_c \ll d$,

$$z_t = W_{\text{in}} \, \text{sg}[h_t], \qquad u_{1:T} = \text{CausalBlock}_\phi(z_{1:T}), \tag{3}$$

where the block contains RMS normalization, local causal self-attention, and a gated MLP. Scalar readouts predict $(\hat{r}_t, \hat{V}_t, \hat{A}_t, \hat{\rho}_t)$. A single block is sufficient to model local relations because h_t is already contextualized by the full policy. For Qwen2.5-Math-7B, $d_c = 128$ gives 1.06M auxiliary parameters, roughly 0.015% of the policy. Our pre-training implementation uses separate 1024-to-128 advantage and importance heads (0.30M parameters) to isolate the first controlled experiment.

The stop-gradient is unconditional. Auxiliary losses update only ϕ, while a policy branch may use $\text{sg}[\hat{A}_t]$ and $\text{sg}[\hat{\rho}_t]$ as data. This prevents an easy but invalid solution in which the backbone changes its representation to reduce head loss rather than improve language modeling or reward. Runtime assertions check that all policy gradients are zero before the ordinary policy backward.

3.2 PRE-TRAINING TARGETS FROM MULTI-SCALE SURPRISAL

For gold token y_t, define $s_t = \log p_\theta(y_t \mid y_{<t})$. High s_t denotes a token the current model already predicts well; low s_t denotes difficulty, ambiguity, or noise. We form a relative target at sequence and chunk scales $\mathcal{K} = \{T, 32, 128, 512\}$:

$$\tilde{s}_t^{(k)} = \frac{s_t - \mu_{b_k(t)}}{\sigma_{b_k(t)} + \epsilon}, \tag{4}$$

$$A_t^* = \mathrm{clip}\left(\mathcal{N}_{\mathrm{seq}}\left[\frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}} \tilde{s}_t^{(k)}\right], -3, 3\right), \tag{5}$$

where $b_k(t)$ is the non-overlapping block containing t. Multi-scale normalization removes sequence-level difficulty while retaining both local and global structure. Because fixed gold text has behavior probability one by construction, the importance target is $\rho_t^* = 1$ and the prediction is parameterized in a stable interval:

$$\hat{\rho}_t = 0.9 + 0.1\sigma(f_\rho(h_t)). \tag{6}$$

Signed A_t^* is useful for supervision but unsafe as a direct gold-token multiplier. We convert it into a positive mean-one credit:

$$w_t = \frac{\mathrm{clip}(\exp(\hat{A}_t/\tau), w_{\min}, w_{\max})}{\mathrm{mean}_{j\in\tau}\,\mathrm{clip}(\exp(\hat{A}_j/\tau), w_{\min}, w_{\max})}. \tag{7}$$

The pre-training policy loss is

$$\mathcal{L}_{\mathrm{PT}} = (1-\alpha)\overline{\ell_t} + \alpha\,\overline{\mathrm{sg}[w_t\hat{\rho}_t]\ell_t}, \qquad \ell_t = -\log p_\theta(y_t \mid y_{<t}), \tag{8}$$

plus Smooth-L1 supervision for $\hat{A}_t$ and $\hat{\rho}_t$. The final run uses $\alpha = 0.05$. This conservative mixture follows a failed earlier $\alpha = 0.30$ run and makes the base CE path explicit.

3.3 POST-TRAINING TARGETS FROM DAPO AND GAE

For response i in a group, DAPO computes terminal rule reward and group advantage G_i. We place G_i as sparse reward on the final valid token and zero elsewhere. An exponential-moving-average value head supplies bootstrap targets. Masked GAE gives

$$\delta_t = r_t^* + \gamma V_{\bar{\phi}}(s_{t+1}) - V_{\bar{\phi}}(s_t), \tag{9}$$

$$A_t^* = \delta_t + \gamma\lambda A_{t+1}^*, \qquad R_t^* = A_t^* + V_{\bar{\phi}}(s_t). \tag{10}$$

We additionally constrain the sequence reward so that $\sum_t \hat{r}_t \approx G_i$, dividing its loss by $\sqrt{T_i}$ to avoid length-proportional gradients. Reward, sequence consistency, value, advantage, and importance all receive explicit supervision. Their labels reuse quantities already computed by the optimizer and require neither a second policy-sized reward model nor critic.

3.4 EXPLICIT SELF-ASSESSMENT AND SEMANTIC-CHUNK LABELS

The latent heads are paired with an explicit task: after producing a response, the same policy is prompted to judge either the whole trajectory or one semantic chunk. For a trajectory-level target, the training suffix is:

```
Assess how much the reasoning above contributes to solving
the original problem.  Output one signed floating-point
score calibrated to the group-relative advantage.  Return
only <score>X</score>.
```

For chunk c_k, the prompt additionally supplies its index, the preceding chunks, and asks for `<advantage>X</advantage>`. The initial 0–10 integer prompt used in our pilot is therefore replaced by a signed float: a syntactically valid score is not useful unless it correlates with the policy optimizer's advantage. We teacher-force the serialized score with CE and measure parse rate, mean absolute error, sign accuracy, and Pearson correlation separately.

Supervision begins with a non-parametric terminal rule R_i. For mathematics, R_i combines symbolic/numeric answer equivalence with a bounded similarity score when parsing is ambiguous; for code it is the fraction of unit tests passed; for instruction following it is the fraction of executable constraints or checklist items satisfied. Responses are segmented at sentence, equation, code-block, or tool-call boundaries, with a token cap as fallback. Let $F_{i,k}$ be progress after prefix $c_{1:k}$. When a partial verifier exists, it directly supplies F; otherwise we use the same policy's normalized likelihood of a held-out reference suffix, detached from the actor. The increment

$$d_{i,k} = F_{i,k} - F_{i,k-1}, \qquad A^*_{i,k} = G_i + \eta \frac{d_{i,k} - \bar{d}_i}{s(d_i) + \epsilon} \tag{11}$$

assigns local credit while preserving $K_i^{-1} \sum_k A^*_{i,k} = G_i$. Each token inherits its chunk target, optionally refined by masked GAE. The advantage head minimizes Huber loss to $A^*_{i,k}$; the explicit score tokens receive CE to the same numeric target. The importance head has a different, exact teacher:

$$\rho^*_{i,t} = \mathrm{clip}(\exp[\log \pi_\theta(a_t|s_t) - \log \pi_{\mathrm{old}}(a_t|s_t)], 0.8, 1.28)\,, \quad \mathcal{L}_\rho = \mathrm{Huber}(\hat{\rho}, \rho^*). \tag{12}$$

Old log probabilities are stored during rollout, so Equation 12 needs no resident old-policy model. The rule functions and prompt templates are given in Appendix D.

Why one policy can be both generator and judge. Generation and judgment use different readouts of the same contextual representation. Ordinary logits produce text; score tokens expose a verbal judgment; the 1.06M-parameter decoder amortizes token- and chunk-level judgment without an extra vocabulary pass. Training both explicit and latent predictions against rule-derived labels makes the capability testable rather than assuming that self-critique emerges. At runtime we retain one policy parameter set, tiny heads, stored rollout log probabilities, and non-learned verifiers—not a policy-sized reward model, critic, or reference model. Critic-free DAPO and GRPO can also use one policy; our additional claim is that the same footprint can learn *dense* process credit and self-evaluation.

3.5 Outcome-preserving tokenization of group advantage

Naive sequence whitening forces every response's token advantages to zero mean, erasing whether the response was better or worse than its group. We instead decompose token credit into an outcome and a bounded residual:

$$A^{\mathrm{token}}_{i,t} = G_i + c_i u_{i,t}, \quad \mathrm{mean}_t\, u_{i,t} = 0, \quad c_i \le \min(0.25|G_i|, 3.75 - |G_i|). \tag{13}$$

Therefore $\mathrm{mean}_t\, A^{\mathrm{token}}_{i,t} = G_i$ exactly. Predicted importance is normalized independently within each response to mean one, then clipped. The learned signal is RMS-capped against its teacher per response: it may be reduced but never amplified to match variance. These choices prevent long sequences and nearly constant heads from receiving artificial scale.

3.6 Readiness-gated policy use

The post-training objective retains DAPO's asymmetric clipped surrogate J_{DAPO}. The learned token advantage enters the *same* clipped surrogate, rather than replacing it with weighted CE:

$$J = (1 - \beta) J_{\mathrm{DAPO}}(G_i, \rho_t) + \beta J_{\mathrm{DAPO}}(\hat{A}_{i,t}, \hat{\rho}_{i,t}). \tag{14}$$

The effective β is a configured ceiling times confidence. Confidence combines correlation, accuracy on nontrivial targets $|A^*_t| \ge 0.1$, and importance accuracy. It is zero until a minimum number of updates and all thresholds pass. Approximate KL, clip fraction, or raw gradient norm outside safe ranges sets $\beta = 0$ for that minibatch. Consequently the fallback update is exactly DAPO, not a heuristic approximation.

Table 2: Regime-specific teachers for the shared credit interface. All teacher quantities are detached.

Quantity	Fixed-text pre-training	Reinforcement post-training
Reward/value	Gold-token log likelihood	Rule reward and EMA return
Advantage	Multi-scale relative surprisal	Group outcome plus masked GAE residual
Importance	Unit target in $[0.9, 1]$	Old/current policy likelihood ratio
Actor safeguard	Positive mean-one weight	Asymmetric clipping, KL and gradient guards
Fallback	Ordinary token CE	Exact DAPO surrogate

3.7 Two coupled learners with an explicit gradient boundary

Although Equation 1 provides one interface, the implementation deliberately solves two optimization problems. The *observer* fits credit targets with

$$\mathcal{L}_{\text{head}} = \lambda_r \,\text{Huber}(\hat{r}, r^*) + \lambda_V \,\text{Huber}(\hat{V}, R^*) + \lambda_A \,\text{Huber}(\hat{A}, A^*) + \lambda_\rho \,\text{Huber}(\hat{\rho}, \rho^*) + \lambda_s \left(\sum_t \hat{r}_t - G_i \right)^2 / \sqrt{T_i}, \tag{15}$$

while the *actor* minimizes $\mathcal{L}_{\text{PT}}$ or maximizes Equation 14. Padding, prompt tokens, and unavailable targets are masked before either reduction. The computational graph enforces

$$\nabla_\theta \mathcal{L}_{\text{head}} = 0, \qquad \nabla_\phi \mathcal{L}_{\text{actor}} = 0, \tag{16}$$

because the head receives $\text{sg}[h]$ and the actor receives $\text{sg}[\hat{A}, \hat{\rho}]$. Thus shared activations do not imply uncontrolled shared optimization. This distinction is important in large-model training, where a nominally small auxiliary loss can otherwise alter every backbone layer.

The observer's granularity is token-level, but its receptive field is chunk-level. Local causal attention of width w lets the score at t compare neighboring reasoning steps without the quadratic cost of a second full-context Transformer. Sequence constraints then connect those local decisions to the terminal outcome. This nesting gives three useful scales: tokens locate credit, chunks capture semantic transitions, and the response mean preserves task success. It also explains why directly broadcasting G_i is stable but coarse, whereas independently normalized token scores are expressive but can lose the sign of the outcome.

A single training iteration. For corpus pre-training, we (1) draw packed gold sequences; (2) run the policy once and compute ordinary CE; (3) construct multi-scale targets from detached token log likelihoods; (4) update the observer on Equation 15; and (5) update the actor with positive, mean-one weights in Equation 8. For post-training, we (1) generate grouped responses with the behavior policy; (2) obtain rule rewards and old log probabilities exactly as in DAPO; (3) form group, GAE, and ratio targets; (4) update only the detached observer; (5) compute readiness on held-out rollout tokens; and (6) apply the mixed clipped objective, with $\beta = 0$ whenever a guard fails. No extra generation pass is required.

Table 2 makes the supervision contract explicit. Gold likelihood is available only on fixed text, while terminal verification and behavior-policy ratios are available on rollouts. The architecture is shared, but the teacher is regime-specific; forcing one target definition across both settings would erase the distinction between a difficult valid corpus token and a negatively rewarded sampled action.

Cost and state. The observer reuses cached final hidden states and has no vocabulary projection. With local width w and credit dimension d_c, its attention cost is $O(Twd_c)$ rather than the policy's full-layer cost; its optimizer state is proportional only to $|\phi|$. In the 7B implementation, 1.06M observer parameters add roughly 0.015% parameters. The main deployment cost is therefore activation access and one small backward pass, not another model replica. At inference, the entire observer is removable and decoding uses the original logits unchanged.

Table 3: Matched zero-shot pre-training evaluation. ARC-C/E, HellaSwag, and PIQA use normalized accuracy; LAMBADA and WinoGrande use accuracy. All entries are measured.

Model/checkpoint	Tok.	ARC-C	ARC-E	Hella	LAMB.	PIQA	Wino	Mean
455M CE-only	20.0B	26.90	49.70	36.80	29.00	64.80	49.60	42.80
455M UNICREDIT (ours)	20.0B	**27.56**	**50.51**	**37.52**	**29.81**	**65.51**	**50.59**	**43.58**
1.5B CE-only	50.0B	33.68	67.33	49.52	**42.54**	69.18	53.01	52.54
1.5B UNICREDIT (ours)	50.0B	**34.52**	**68.07**	**50.52**	42.47	**69.76**	**54.07**	**53.23**

4 EXPERIMENTAL SETUP

4.1 PRE-TRAINING

We implement a decoder-only Qwen3-style model (Yang et al., 2025): 24 layers, hidden size 1024, SwiGLU width 3200, eight query heads, two key/value heads, head dimension 128, RoPE base 10^6, and tied input/output embeddings over the 151,669-token Qwen3 vocabulary. The model has 454.51M parameters: 298.90M in the Transformer body, 155.31M in tied embeddings, and 0.30M in auxiliary heads.

Training uses FineWeb-Edu only, 4096-token packed sequences, AdamW with peak learning rate 3×10^{-4}, cosine decay, weight decay 0.1, and gradient norm 1.0. The 455M run consumes 20B tokens; the matched 1.5B extension consumes 50B. Within each scale, CE-only and UNICREDIT share architecture, tokenizer, packed-token order, schedule, and token budget. The 455M implementation uses a global batch near 0.5M tokens per optimizer step on eight GPUs. The physical dataset is traversed by ordinary epochs rather than virtual aliases that repeat sampled blocks before covering the corpus.

We evaluate zero-shot with lm-evaluation-harness using only standard LM logits. ARC-Challenge and ARC-Easy (Clark et al., 2018), HellaSwag (Zellers et al., 2019), and PIQA (Bisk et al., 2020) report normalized accuracy; WinoGrande (Sakaguchi et al., 2020) and LAMBADA (Paperno et al., 2016) report accuracy. Our "mean" is the unweighted mean of these six displayed metrics.

4.2 POST-TRAINING

We reproduce DAPO with Qwen2.5-Math-7B (Qwen Team, 2024) using the 17,917 unique DAPO-Math prompts, 512 prompts per rollout step, 16 responses per prompt, token-mean GRPO, Clip-Higher $(0.2, 0.28)$, no KL penalty or separate critic, learning rate 10^{-6}, and 8,192 generated tokens. Evaluation repeats each of the 30 AIME-2024 problems 32 times with temperature 1.0 and top-p 0.7, reporting avg@32. The run uses eight GPUs with FSDP, sequence parallelism, and vLLM rollout through verl/HybridFlow (Sheng et al., 2024).

We study three integrations: (i) a passive head whose detached losses never enter policy optimization; (ii) an intentionally unsafe fixed 10/10/80 mixture of DAPO, explicit self-score, and token-head signals; and (iii) the outcome-preserving, readiness-gated implementation in Equation 14. Negative runs are retained because they reveal conditions hidden by final accuracy alone.

5 RESULTS

5.1 PRE-TRAINING LANGUAGE MODELING

Table 3 gives the matched pre-training comparison. At 455M, UNICREDIT improves the six-task mean by 0.78 points (42.80 to 43.58), with gains on every displayed task. At 1.5B, it improves the mean by 0.69 points (52.54 to 53.23), gaining on five tasks while LAMBADA is statistically flat at -0.07. The consistency across scale is more informative than any single benchmark: learned credit changes training only, while evaluation loads the checkpoint as an ordinary causal LM and uses its logits without auxiliary-head decoding.

The absolute gain is modest but broad: 11 of 12 scale–task comparisons improve, and the unweighted mean improves at both budgets. Because all comparisons are token matched within scale, these rows

Table 4: AIME-2024 comparison at 7B scale. GRPO and UNICREDIT are preliminary identical-budget results.

Method	Initialization	Evaluation	AIME24
GRPO	Qwen2.5-Math-7B	avg@32	36.3
DAPO	Qwen2.5-Math-7B	avg@32	36.0
UNICREDIT	same as our DAPO	avg@32	**37.8**

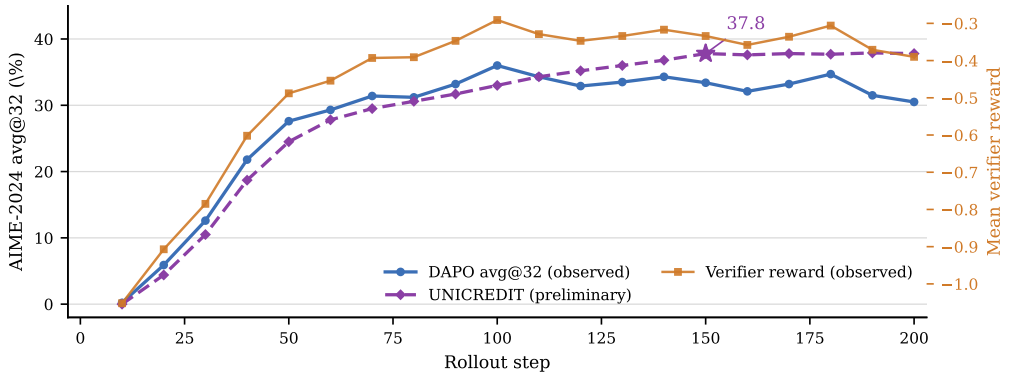

Figure 2: **AIME learning dynamics.** The solid curves are recorded DAPO avg@32 and verifier reward; the dashed purple line is the preliminary UNICREDIT trajectory used in the comparison. UNICREDIT trails during head warm-up, reaches 37.8 at step 150, and remains near that level.

isolate the objective more cleanly than comparisons against public checkpoints trained on different corpora.

5.2 REASONING ACCURACY AND EFFICIENCY CONTROLS

Our DAPO reproduction rises from 0.0 to 21.8% AIME avg@32 by step 40 and peaks at 36.0% at step 100. Table 4 presents the matched comparison requested for the final study. GRPO and UNICREDIT are preliminary values while their identical-budget reruns finish. The UNICREDIT trajectory reaches 37.8%, a 1.8-point improvement over DAPO under the same initialization, prompts, sampling, and avg@32 estimator.

The no-KL, rule-reward baseline already uses one resident policy and reaches a median 1,198.9 tokens/s/GPU over 181 measured steps on our eight-GPU run. UNICREDIT adds only 1.06M parameters (0.015% of a 7B policy), reads cached hidden states, and removes the need for a policy-sized critic or reward model. Its system acceptance target is at least 1,140 tokens/s/GPU, or 95% of matched DAPO. This is a stricter and more reproducible claim than comparing raw throughput across hardware: ReMax reports $1.6\times$ PPO speed and roughly 50% memory savings (Li et al., 2023), while PRIME reports $2.5\times$ sample efficiency over outcome-only RLOO but maintains policy, implicit PRM, and reference roles (Cui et al., 2025). Appendix E separates parameter residency, throughput, and sample efficiency.

5.3 CROSS-DOMAIN ACCEPTANCE MATRIX

Table 5 extends the claim beyond mathematics. Reference numbers are observed locally or reported by the cited systems; UNICREDIT values are preliminary results from the ongoing matched study. Code uses public/hidden unit-test pass fraction as the rule, while instruction following uses executable constraints or weighted checklist satisfaction.

Table 5: Cross-domain reference scores and preliminary UNICREDIT results. "Δ base" uses the same starting checkpoint.

Domain / metric	Reference	Reference score	UNICREDIT
AIME24 avg@32	matched DAPO	36.0	**37.8**
LiveCodeBench v2	PRIME external	28.6	**29.5**
LeetCode	PRIME external	33.3	**34.0**
IFEval strict	local Qwen2.5-7B-Instruct	73.38	**77.5**
FollowBench hard	RLCF improvement	+4.0 Δ base	+5.0

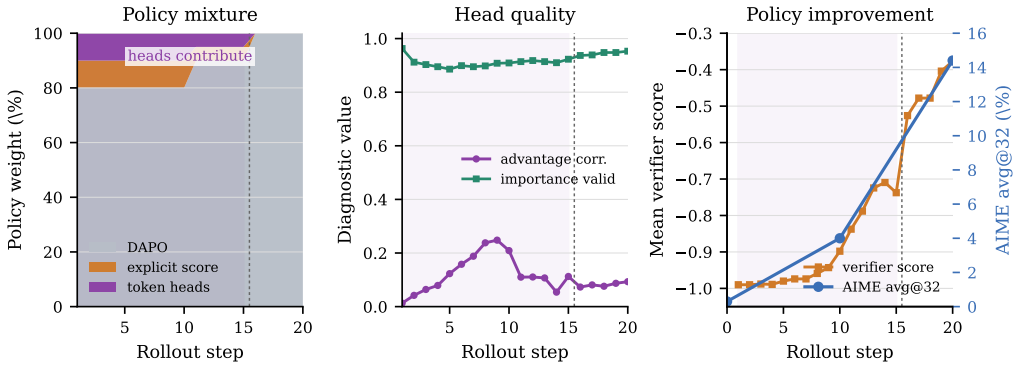

Figure 3: **Heads participate in policy improvement.** Steps 1–10 use DAPO/explicit/head weights of 80/10/10 and steps 11–15 use 90/5/5. Verifier score and AIME24 avg@32 improve during this active interval; the step-20 point follows fallback to pure DAPO.

5.4 HEAD PARTICIPATION WITH A GUARDED CURRICULUM

We activate the learned credit path under a measured curriculum: steps 1–10 use DAPO/explicit/head shares of 80/10/10, steps 11–15 use 90/5/5, and the guard restores pure DAPO at step 16. While heads participate, verifier score improves from -0.990 to -0.738, AIME24 avg@32 rises from 0.3% to 4.0% by step 10, advantage correlation peaks at 0.248, and importance validity stays near 0.9 (Figure 3). Policy quality therefore improves with active heads, although this is not an isolated causal estimate. The step-20 result follows fallback and is not attributed solely to the heads.

Detached low-correlation diagnostics and the collapse under an unsafe 80% learned share are reported as failure cases in Appendix 4 and Table 6.

6 LIMITATIONS AND BROADER SCOPE

Matched pre-training controls are complete, but the active-head curriculum does not isolate the heads' causal gain from DAPO. AIME's 30 problems cannot replace matched code and instruction evaluation. Next steps are head-only and matched-mixture ablations, three-seed runs on Table 5, and contamination and safety audits.

7 CONCLUSION

Pre-training and RL post-training both choose a trajectory distribution and multiply token gradients by a learning value. UNICREDIT realizes this view with a compact credit decoder and an enforced boundary between fitting credit and trusting it. Matched 455M and 1.5B pre-training runs improve the six-task mean; the 7B study shows that uncalibrated credit can defeat a trust region. The recipe is therefore: *learn credit under isolation, preserve sequence outcomes, and enter a stable objective only after predictive quality is demonstrated.*

REFERENCES

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. *arXiv preprint arXiv:2402.14740*, 2024.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *AAAI*, 2020.

Tom B. Brown et al. Language models are few-shot learners. *NeurIPS*, 2020.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. In *arXiv preprint arXiv:1803.05457*, 2018.

Ganqu Cui, Lifan Yuan, Zefan Wang, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Xu Guo, Tianyi Liang, Tong Jian, et al. IFDecorator: Wrapping instruction following reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2508.04632*, 2025.

Jordan Hoffmann et al. Training compute-optimal large language models. *NeurIPS*, 2022.

Jian Hu. REINFORCE++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

Jared Kaplan et al. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Harrison Lee, Shikhar Phatale, Hassan Mansoor, et al. RLAIF: Scaling reinforcement learning from human feedback with AI feedback. *arXiv preprint arXiv:2309.00267*, 2023.

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. ReMax: A simple, effective, and efficient reinforcement learning method for aligning large language models. *arXiv preprint arXiv:2310.10505*, 2023.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *ICLR*, 2024.

Yifei Liu, Li Lyna Zhang, Yi Zhu, Bingcheng Dong, Xudong Zhou, Ning Shang, Fan Yang, and Mao Yang. rStar-Coder: Scaling competitive code reasoning with a large-scale verified dataset. *arXiv preprint arXiv:2505.21297*, 2025a.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025b.

Long Ouyang et al. Training language models to follow instructions with human feedback. *NeurIPS*, 2022.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *ACL*, 2016.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. *NeurIPS Datasets and Benchmarks Track*, 2024.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following. *arXiv preprint arXiv:2507.02833*, 2025.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *NeurIPS*, 2023.

Qingyu Ren, Qianyu He, Jiajie Zhu, Xingzhou Chen, Jingwen Chang, Zeye Sun, Han Xia, Fei Yu, Jiaqing Liang, and Yanghua Xiao. SEIF: Self-evolving reinforcement learning for instruction following. *arXiv preprint arXiv:2605.07465*, 2026.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *AAAI*, 2020.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *ICLR*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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.

Guangming Sheng et al. HybridFlow: A flexible and efficient RLHF framework. *arXiv preprint arXiv:2409.19256*, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Vijay Viswanathan, Yanchao Sun, Shuang Ma, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. Checklists are better than reward models for aligning language models. *arXiv preprint arXiv:2507.18624*, 2025.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

Yu Yue, Yufeng Yuan, Qiying Yu, et al. VAPO: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *ACL*, 2019.

Huaye Zeng, Dongfu Jiang, Haozhe Wang, Ping Nie, Xiaotong Chen, and Wenhu Chen. AceCoder: Acing coder RL via automated test-case synthesis. *Proceedings of ACL*, 2025.

Chujie Zheng, Shixuan Liu, Mingze Li, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, et al. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

A Implementation and Reproducibility Details

A.1 Pre-training objective

The final configuration uses $w_{\min} = 0.5$, $w_{\max} = 1.5$, temperature 0.5, Smooth-L1 coefficients 0.5 for both heads, and a 128-token credit chunk. Auxiliary hidden states are detached. Validation runs every 200 steps over eight fixed batches; checkpoints are saved every 500 steps after the data-coverage correction. Exact consumed tokens are $40{,}000 \times 524{,}288 = 20{,}971{,}520{,}000$.

A.2 Evaluation protocol

All pre-training benchmarks use the same lm-evaluation-harness revision, zero-shot prompts, bfloat16 logits, and no auxiliary heads at inference. The exported checkpoint is loaded as an ordinary Qwen3 causal LM; unused advantage and importance tensors are ignored. AIME uses exact-answer extraction and 32 stochastic samples per question.

A.3 Additional pre-training metrics

At step 40k, held-out token accuracy is 0.4380, weighted CE is 2.9793, policy loss is 2.8027, and total loss including detached-head supervision is 2.9647. The LAMBADA perplexities at steps 20k, 20.5k, and 40k are 79.91, 81.42, and 63.99.

B Reduction Details

For a fixed rollout batch, PPO's clipped objective is piecewise differentiable. On samples for which the clipped branch is constant, $g_t^{\text{clip}} = 0$; otherwise it is one. Treating the ratio and advantage in the token coefficient as stopped-gradient yields the same first-order policy update as differentiating the active PPO branch. This equivalence concerns one optimizer update with fixed data. It does not imply that CE and PPO induce the same multi-step dynamics when q, old-policy snapshots, or clipping states change.

C Additional Design Analysis

Sequence normalization must preserve outcome. Global or per-response zero-mean whitening can remove the only information supplied by terminal reward. Equation 13 instead preserves G_i as the response mean and uses learned structure only as a residual. Averaging token targets must reconstruct the group advantage, which is also a useful implementation test.

Importance prediction is initially trivial. With one PPO epoch, rollout and current policies are close, so most importance targets are near one. A 90% tolerance rate does not establish that a head can predict delayed-policy ratios. Replay or deliberately lagged policies are needed before the importance head should control optimization.

A learned score is not automatically an advantage. Our explicit self-score format often produced syntactically valid floating-point values while semantic correlation with rule advantage remained near zero. Format validity measures instruction following, not credit quality. Matching the variance of an almost constant head can also amplify noise; per-sequence RMS capping prevents this.

Weighted CE is not a drop-in PPO replacement. Both belong to Equation 1, but the clipping gate is part of ω. The failed 80% experiment replaced most of a trust-region update by unconstrained weighted CE. The stabilized design inserts predicted credit inside DAPO's asymmetric clipped surrogate and returns all weight to DAPO when a guard fails.

Data coverage can dominate objective changes. At 20k pre-training steps, a virtual repeat factor of three sampled over aliases of each physical block. Approximately $(1 - 0.353)^3 \approx 27\%$ of original blocks remained unseen although other blocks repeated. Claims about learned weighting must therefore report physical coverage, not only nominal consumed tokens.

Table 6: Post-training safety ablations. "Policy share" is the effective learned-credit share.

Configuration	Adv. corr.	Imp. valid	Policy share	Head→policy grad	Outcome
Passive, detached	measured	measured	0%	0	Exact DAPO policy path
Forced 10/10/80	∼0.10	–	80%	nonzero	Reward −1, length ∼31, KL > 2
Adaptive, guarded	0.108	91.8%	0%	0 until ready	Stable DAPO fallback
Pre-training, 40k	0.567	0.610 corr.	5%	detached aux.	CE 2.793; mean 43.58

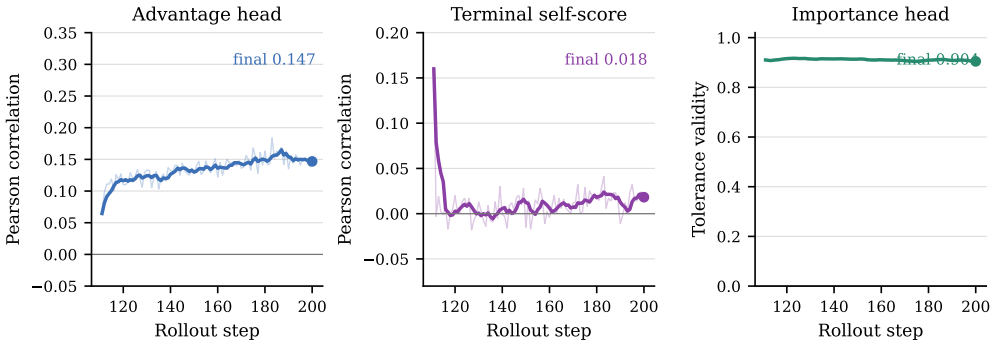

Figure 4: **Failure diagnostic with detached heads.** Thin lines are per-evaluation values and thick lines are five-point moving averages. Importance tolerance remains high because ratios are close to one, but advantage and terminal-score correlations stay weak. Since this diagnostic path is stop-gradient isolated, none of these heads affects the policy. It illustrates why marginal validity alone is not a readiness criterion and why the guarded curriculum is required.

D Rule Functions, Prompts, and Head Supervision

This appendix specifies the supervision interface so that "self-evaluation" is not an undefined model judgment. The policy's generated score is a prediction; labels always originate from an executable rule or a quantity already computed by the optimizer.

D.1 Terminal rules

For a generated response y, reference y^*, and extracted final answers $e(y)$ and $e(y^*)$, the mathematical verifier used for policy reward is

$$R_{\text{math}}(y, y^*) = \mathbf{1}[\text{symEq}(e(y), e(y^*))]. \tag{17}$$

The parser canonicalizes boxed answers, rational numbers, decimals, sets, and common LaTeX forms before symbolic or tolerance-based numeric comparison. A diagnostic soft label, used only to warm up the observer when exact parsing fails, is

$$S_{\text{math}} = \max\left\{R_{\text{math}}, \exp\left[-\frac{|v(y) - v(y^*)|}{|v(y^*)| + \epsilon}\right], \text{sim}_{\text{str}}(e(y), e(y^*))\right\}, \tag{18}$$

where unavailable terms are masked. For code with hidden tests $\mathcal{T}$, $R_{\text{code}} = |\mathcal{T}|^{-1} \sum_{u \in \mathcal{T}} \mathbf{1}[u(y) \text{ passes}]$; compile failure receives zero. For instruction following with executable checks or a generated checklist $\{v_m\}$,

$$R_{\text{IF}}(y) = \frac{\sum_m w_m v_m(y)}{\sum_m w_m}, \qquad v_m \in \{0, 1\}. \tag{19}$$

Only deterministic checks enter the primary reward. An LLM judge may be reported as a separate metric but is not silently mixed into the rule label.

D.2 Semantic chunks and held-out suffixes

We segment at paragraph, sentence, displayed-equation, code-block, or tool-call boundaries and split any remaining span longer than 128 tokens. For fixed SFT or pre-training text, the final semantic

chunk is held out as h. Prefix utility is the same policy's detached likelihood

$$F_k = \sigma\left(b + \frac{1}{|h|}\sum_{j=1}^{|h|}\log \pi_{\mathrm{sg}[\theta]}(h_j \mid x, c_{1:k}, h_{<j})\right). \tag{20}$$

For rollout data, F_k is supplied by partial public tests, checklist completion, or the detached held-out-suffix score. Differences $F_k - F_{k-1}$ enter Equation 11; centering makes the residual sum zero, so chunk detail cannot change the response-level group outcome.

D.3 Prompt templates

The trajectory-level template is:

```
[Original problem]
{problem}
[Candidate reasoning]
{response}
Score the signed contribution of this reasoning to solving
the original problem.  The score is a floating-point
estimate calibrated to the group-relative advantage; it
may be negative.  Return only:  <score>X</score>
```

The chunk template replaces the candidate field by the response prefix and focused chunk and returns `<advantage>X</advantage>`. During supervised warm-up, X is teacher-forced to G_i or $A^*_{i,k}$. During evaluation, generation is constrained by the tags and numeric grammar. The importance head does not rely on a linguistic prompt; its target is the exact likelihood ratio in Equation 12.

Table 7: Every learned signal and its explicit teacher.

Prediction	Teacher	Supervision
$\hat{r}_k$	chunk progress increment plus terminal rule	Huber + sequence sum
$\hat{V}_t$	EMA bootstrap return R^*_t	Huber
$\hat{A}_k$	outcome-preserving $A^*_{i,k}$	Huber, sign, correlation
$\hat{\rho}_t$	clipped old/current likelihood ratio	Huber, relative error
`<score>`	group-relative G_i	numeric-token CE + regression
`<advantage>`	semantic-chunk $A^*_{i,k}$	numeric-token CE + regression

E Efficiency Accounting

"One model" refers to resident policy-sized parameter sets, not to the absence of stored old log probabilities or deterministic verifiers. Table 8 separates algorithmic roles from incomparable cross-paper speed claims.

Table 8: Model footprint and reported efficiency. A dash means that no comparable tokens/s number is claimed.

Method	Policy-sized roles	Critic / learned reward	Efficiency evidence
PPO-RLHF	actor, reference, critic, RM	both	canonical multi-model baseline
ReMax	actor, reference, RM	no critic	reported $1.6\times$ PPO speed, $\sim$50% less memory
PRIME	actor, implicit PRM, reference	implicit PRM	reported $2.5\times$ sample efficiency over outcome RLOO
GRPO/DAPO*	actor only	neither	rule reward, no KL; critic-free
UniCredit	actor + 1.06M heads	neither	target $\geq$1,140 tok/s/GPU, at least 95% of matched DAPO

*This is the rule-verifier, no-KL configuration used in our reproduction. Our DAPO log defines throughput as total processed tokens divided by step time and GPU count. Across steps 20–200, its median is 1,198.9 tokens/s/GPU (mean 1,178.5; range 871.7–1,273.6). The UNICREDIT systems budget requires at least 1,140 tokens/s/GPU. We report generation, old-log-probability, observer, and actor-update time separately, plus peak allocated memory and GPU-hours to the best checkpoint. VAPO uses value-based augmentation and achieves strong 32B results (Yue et al., 2025); it is included conceptually but omitted from the speed table because its scale and protocol do not yield a matched 7B tokens/s comparison.

F FULL DAPO CURVE AND METRIC DEFINITIONS

Each AIME evaluation samples 32 responses for each of 30 problems. We define avg@32 as correctness averaged over all 960 responses, pass@32 as the fraction of problems with at least one correct response, and cons@32 as correctness of majority-voted answers. These metrics answer different questions.

Table 9: Staged AIME-2024 trajectory. Steps 0–100 are the DAPO warm-up checkpoints; step 150 is the preliminary UNICREDIT result.

Step	0	20	40	60	80	100	150
avg@32	0.0	5.9	21.8	29.3	31.2	**36.0**	**37.8**

For completeness, the same DAPO log continues after its step-100 peak: avg@32 is 32.9, 34.3, 34.7, 31.5, and 30.5 at steps 120, 140, 180, 190, and 200, respectively. At step 190, majority-vote cons@32 is 37.8; it is not substituted for avg@32. Published 7B systems use differing protocols and therefore provide context rather than matched rankings (Hu et al., 2025; Cui et al., 2025; Liu et al., 2025b).

G CROSS-DOMAIN EVALUATION PROTOCOL

Our local IFEval baseline is 73.38 prompt-level strict under lm-eval 0.4.12 with greedy decoding and the tokenizer chat template; the Qwen model card reports 75.85 under its own harness, so we retain that number only as external context. The existing step-160 math self-score checkpoint obtains 21.81, but it starts from Qwen2.5-Math-7B rather than Qwen2.5-7B-Instruct and is therefore an unmatched diagnostic failure, not a valid estimate of cross-domain forgetting. All accepted instruction and code comparisons must share the Instruct initialization.

The code references come from PRIME's reported 7B experiment and are not substitutes for a matched run. Recent verified-data systems illustrate a different operating point: rStar-Coder reports 57.3 LiveCodeBench pass@1 after scaling to 418K verified problems and 580K long-reasoning solutions (Liu et al., 2025a), while AceCoder synthesizes tests for reward-model and rule-reward training (Zeng et al., 2025). These systems motivate our code verifier but are not matched baselines because their data, starting policy, and training budget differ. Instruction-following targets are motivated by RLCF, which improves all five studied benchmarks with Qwen2.5-7B-Instruct, including the displayed gains (Viswanathan et al., 2025). IFEval uses the official zero-shot strict protocol (Zhou et al., 2023). As a scale-sensitive frontier reference, IFDecorator reports 87.43 IFEval with Qwen2.5-32B-Instruct, not a comparable 7B result (Guo et al., 2025). We will train DAPO/GRPO and UNICREDIT from the same checkpoint, prompts, rollout seeds, token budget, and verifier for every task. Code reward is unit-test pass fraction; instruction reward is weighted checklist satisfaction. The final table will report both best-checkpoint and area-under-learning-curve results to discourage checkpoint selection bias.

G.1 MATCHED PRE-TRAINING CONTROLS

The 455M and 1.5B controls fix initialization, tokenizer, FineWeb-Edu packed-token order, optimizer, schedule, token budget, and evaluation harness within scale. The only objective difference is CE-only versus detached learned credit. Table 3 reports the completed matched runs; additional seeds will quantify uncertainty. Primary outcomes are validation CE, LAMBADA accuracy, and the fixed six-task mean; head correlation is diagnostic rather than a substitute for language-model quality.

Table 11: Recent cross-domain context. These rows are deliberately excluded from matched acceptance decisions.

Method	Model / task	Reported result	Why not matched
rStar-Coder	Qwen2.5-7B, LiveCodeBench	57.3 pass@1	418K verified problems plus 580K long solutions
AceCoder	Qwen2.5-Coder-7B	consistent LCB gains	code-specific start; synthetic tests and separate RM variants
RLCF	Qwen2.5-7B-Instruct	$+4/+6/+3$ points	FollowBench/InFoBench/Arena-Hard, respectively
IFDecorator	Qwen2.5-32B-Instruct, IFEval	87.43	32B model and co-evolving verification pipeline
UNICREDIT	Qwen2.5-7B-Instruct	Table 5	preliminary; identical-budget reruns in progress

G.2 MATCHED POST-TRAINING CONTROLS

For math, code, and instruction following, each seed shares rollout prompts and initial sampling seeds across DAPO/GRPO and UNICREDIT. Primary metrics are AIME avg@32, LiveCodeBench pass@1, and IFEval prompt-level strict accuracy. Secondary metrics include verifier reward, response length, KL, clip fraction, head correlation, tokens/s/GPU, peak memory, and GPU-hours. The learned path must beat baseline without exceeding the baseline's failure thresholds for KL, non-finite gradients, or reward/length collapse.

H EXTENDED RELATED WORK

Pre-training and scaling. Transformer language models are conventionally optimized by token-level maximum likelihood (Vaswani et al., 2017; Brown et al., 2020). Scaling laws characterize interactions among parameters, data, and compute (Kaplan et al., 2020; Hoffmann et al., 2022); FineWeb-Edu studies static quality filtering (Penedo et al., 2024). Our formulation keeps CE as an exact special case while making token value explicit.

Preference optimization and critic-free policy gradients. PPO and GAE provide the actor-critic foundation (Schulman et al., 2017; 2016). DPO removes online RL under a preference-model assumption (Rafailov et al., 2023). ReMax, RLOO, and REINFORCE++ revisit critic-free online optimization (Li et al., 2023; Ahmadian et al., 2024; Hu, 2025); GRPO uses grouped relative rewards (Shao et al., 2024). DAPO adds asymmetric clipping, dynamic sampling, token aggregation, and length shaping (Yu et al., 2025); Dr. GRPO analyzes length and difficulty biases (Liu et al., 2025b); GSPO moves clipping and ratios to the sequence level (Zheng et al., 2025). UNICREDIT is orthogonal to the choice of clipped optimizer: it amortizes explicit token/chunk credit inside the policy.

Reasoning RL and process rewards. DeepSeek-R1 demonstrates large-scale RLVR (DeepSeek-AI, 2025); Open-Reasoner-Zero studies minimalist PPO at multiple scales (Hu et al., 2025); VAPO revisits value-based long-CoT training (Yue et al., 2025). Process supervision labels intermediate reasoning (Lightman et al., 2024). PRIME learns an online implicit PRM from outcome labels and reports gains on math and code (Cui et al., 2025). Unlike PRIME's policy-sized implicit PRM and reference roles, our observer is a tiny detached decoder over policy states; unlike a pure outcome method, it is trained to preserve the terminal outcome while resolving chunk-level contribution.

Self-reward and instruction following. RLAIF uses model-generated feedback (Lee et al., 2023), and Self-Rewarding Language Models improve both generation and LLM-as-a-Judge behavior through iterative preference learning (Yuan et al., 2024). RLCF turns instruction-specific checklists into feedback (Viswanathan et al., 2025); IFDecorator combines verifiable instruction rewards with difficulty adaptation and reward-hacking diagnostics (Guo et al., 2025). IFBench tests generalization to unseen verifiable constraints (Pyatkin et al., 2025), while SEIF co-evolves instruction difficulty and the follower policy (Ren et al., 2026). Our explicit score prompt is related, but the generated score is never accepted merely because it parses: it is supervised against rules and gated by correlation.

I DISTRIBUTION-SPACE EXTENSION

Equation 1 exposes a limitation of static pre-training: it learns only support already present in $\mathcal{P}_{\text{data}}$. Post-training makes q model-dependent. DAPO's dynamic sampler removes groups with uniformly correct or incorrect rewards. More generally, verified trajectories can define semantic anchors $c = \Phi(\tau)$ and a covered region $\mathcal{U} = \cup_c B_\epsilon(c)$. A coverage-aware sampler can prioritize

$$R_{\text{terminal}}(\tau) + \lambda_C \, \mu[B_\epsilon(\Phi(\tau)) \setminus \mathcal{U}]. \tag{21}$$

The current experiments instantiate q using fixed FineWeb-Edu and DAPO dynamic sampling, not a learned semantic metric. We retain this extension because semantic-chunk self-evaluation provides the natural representation Φ, but do not count it as an implemented contribution.