Overconfident Errors Need Stronger Correction: Asymmetric Confidence Penalties for Reinforcement Learning

Anonymous authors
Paper under double-blind review

Abstract

Reinforcement Learning with Verifiable Rewards (RLVR) has become the leading paradigm for enhancing reasoning in Large Language Models (LLMs). However, standard RLVR algorithms suffer from a well-documented pathology: while improving Pass@1 through sharpened sampling, they simultaneously narrow the model's reasoning boundary and reduce generation diversity. We identify a root cause that existing methods overlook: the *uniform penalization of errors*. Current approaches—whether data-filtering methods that select prompts by difficulty, or advantage normalization schemes—treat all incorrect rollouts within a group identically. We show that this uniformity allows *overconfident errors*—incorrect reasoning paths that the RL process has spuriously reinforced—to persist and monopolize probability mass, suppressing valid exploratory trajectories.

We propose the **Asymmetric Confidence-aware Error Penalty (ACE)**, which introduces a per-rollout confidence shift metric $c_i = \log(\pi_\theta(y_i|x)/\pi_{\text{ref}}(y_i|x))$ to dynamically modulate negative advantages. Theoretically, we show that ACE's gradient can be decomposed into the gradient of a *selective regularizer* restricted to overconfident errors, plus a well-characterized residual that partially moderates the regularizer's strength (Theorem 1). Experiments fine-tune Qwen2.5-Math-7B (Qwen Team et al., 2024), Qwen3-8B-Base (Yang et al., 2025), and Llama-3.1-8B-Instruct (Grattafiori et al., 2024) on the DAPO-Math-17K dataset (Yu et al., 2025) using GRPO and DAPO with VERL (Volcano Engine, 2024), evaluating on MATH-500 (Hendrycks et al., 2021) and AIME 2025. ACE yields the strongest and most consistent gains on the two Qwen families, and ACE-GRPO also improves large-k performance on Llama-3.1-8B-Instruct, indicating transfer beyond the primary model family.

1 Introduction

Reinforcement Learning with Verifiable Rewards (RLVR) (DeepSeek-AI et al., 2026; OpenAI et al., 2024) has emerged as a primary method for post-training Large Language Models (LLMs) on reasoning tasks. By using binary correctness signals from deterministic verifiers, algorithms such as PPO (Schulman et al., 2017), GRPO (Shao et al., 2024), and REINFORCE (Williams, 1992) iteratively refine the model's Chain-of-Thought (CoT) generation (Wei et al., 2022).

Despite its successes, a growing body of evidence reveals a fundamental tension in RLVR training. While RLVR models excel at Pass@1, they consistently underperform their own base models at Pass@k for large k (Chen et al., 2021; Yue et al., 2025; Brown et al., 2024), indicating a *narrowing* of the reasoning boundary rather than an expansion. This phenomenon has been attributed to diversity collapse: the training process concentrates probability mass on a small number of successful reasoning paths, suppressing the broader solution space.

A prominent strategy addresses this: Difficulty-based curriculum learning filters prompts to maximize gradient signal. However, such methods operate at a macro level—selecting which problems to train on—ignoring a critical micro-level distinction: not all errors are equal.

Within incorrect rollouts, we identify distinct regimes: *exploratory errors* (benign stochastic deviations), *self-correcting errors* (paths the model is already abandoning), and *overconfident errors* (spuriously reinforced

paths acting as value traps). Standard RLVR penalizes these uniformly. While the global KL penalty $\beta\mathbb{D}_{\mathrm{KL}}(\pi_\theta\|\pi_{\mathrm{ref}})$ offers some correction, it is symmetric and indiscriminate, suppressing beneficial exploration alongside harmful overconfidence.

Our contribution. We propose to break this dilemma by introducing *asymmetric* correction at the level of individual rollouts. Our method, **ACE** (**A**symmetric **C**onfidence-aware **E**rror penalty), dynamically amplifies the penalty for overconfident errors using a per-rollout confidence shift metric, while leaving exploratory and self-correcting errors largely untouched. Concretely, our contributions are:

1. **A new analytical dimension.** We formalize *error confidence shift* $c_i = \log(\pi_\theta(y_i|x)/\pi_{\mathrm{ref}}(y_i|x))$ as a per-rollout diagnostic that is orthogonal to prompt-level difficulty, and show empirically that overconfident errors accumulate during training (§3).
2. **Theoretical foundations.** We show that ACE's gradient admits a decomposition into a *selective regularizer* targeting the overconfident portion of the policy, plus a residual term that partially moderates the regularizer's correction strength (§4.4).
3. **Empirical validation.** ACE delivers the strongest and most consistent improvements on the two Qwen families across both GRPO and DAPO, with particularly strong gains at large k, and ACE-GRPO also improves large-k performance on Llama-3.1-8B-Instruct, supporting transfer beyond the primary model family (§5).

Figure 1 highlights the core idea: ACE reshapes the negative penalty as a smooth, confidence-dependent curve, amplifying penalties for overconfident errors while keeping exploratory and self-correcting errors close to the base level.

2 Related Work

Curriculum and advantage shaping. Curriculum methods (Zeng et al., 2025; Parashar et al., 2025; Zhang et al., 2025) select prompts by difficulty, operating at the *prompt level*. Advantage shaping methods (Tang et al., 2025; Wen et al., 2025) balance correct vs. incorrect samples at the *group level*. ACE operates at the *rollout level*, modulating penalties within incorrect samples based on per-rollout confidence shift.

KL regularization in RLHF/RLVR. KL divergence penalties are standard in RLHF pipelines to prevent reward hacking and mode collapse (Ouyang et al., 2022; Stiennon et al., 2020). The typical formulation adds a global term $\beta\mathbb{D}_{\mathrm{KL}}(\pi_\theta\|\pi_{\mathrm{ref}})$ that symmetrically penalizes all deviations from the reference. DPO (Rafailov et al., 2023a) implicitly constrains the KL divergence through its closed-form reward parameterization. However, all these methods apply KL penalties *uniformly* across correct and incorrect outputs alike, suppressing beneficial exploration alongside harmful overconfidence. ACE introduces an *asymmetric* and *selective* KL-like penalty that targets only overconfident errors while leaving correct outputs and self-correcting errors untouched.

Entropy regularization and clipping strategies. Entropy bonuses have a long history in RL for encouraging exploration (Williams, 1992; Schulman et al., 2017). In the LLM context, DAPO (Yu et al., 2025) combats entropy collapse through its Clip-Higher strategy, which decouples the upper and lower clipping thresholds of the importance sampling ratio to give low-probability exploration tokens more room for probability increase. While such clipping-based strategies promote diversity globally, they operate at the *token level* and cannot distinguish between beneficial diversity on correct reasoning paths and harmful persistence of incorrect ones. ACE provides a complementary, more targeted mechanism: rather than modifying the clipping bounds, it modulates the penalty magnitude per *rollout* based on confidence shift, achieving diversity preservation as a *consequence* of selectively suppressing overconfident errors (see §5.4).

Reward shaping. Potential-based reward shaping (Ng et al., 1999; Wiewiora et al., 2003; Devlin & Kudenko, 2012) transforms the reward function to accelerate learning while preserving the optimal policy. ACE can be viewed through the reward shaping lens: the confidence-dependent term $\alpha \cdot \mathrm{Softplus}(c_i)$ acts

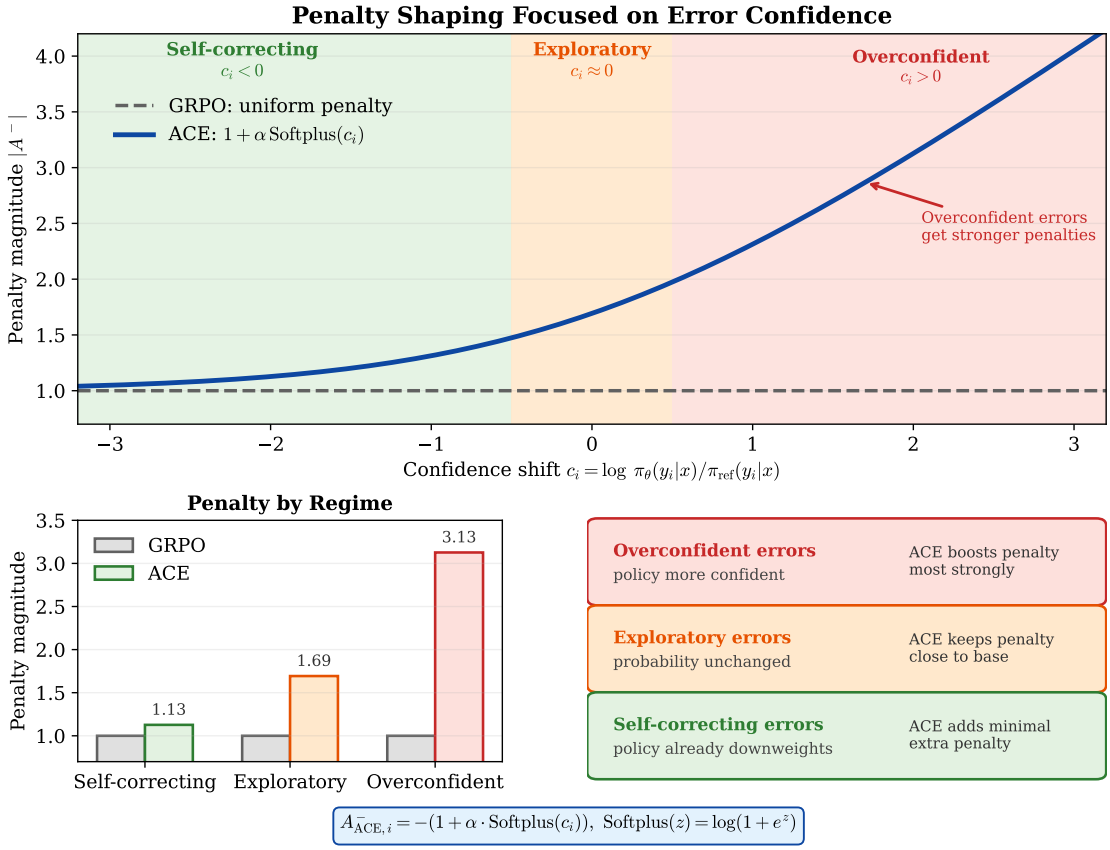

Figure 1: **ACE Method Overview.** *Top:* Incorrect rollouts fall into three regimes based on the confidence shift $c_i = \log(\pi_\theta(y_i|x)/\pi_{\mathrm{ref}}(y_i|x))$. *Bottom-left:* Standard GRPO assigns a uniform penalty $|\hat{A}^-|$ to all errors regardless of regime. *Bottom-right:* ACE modulates the penalty via $\mathrm{Softplus}(c_i)$, strongly penalizing overconfident errors while leaving self-correcting errors nearly untouched.

as an auxiliary reward signal derived from the policy–reference divergence. Unlike classical potential-based shaping, ACE's shaping signal is *asymmetric* (applied only to negative advantages) and *adaptive* (it evolves with the policy). Process reward models (Lightman et al., 2024) offer another form of reward enrichment at the step level; ACE is complementary, operating at the trajectory level with zero additional annotation cost.

Diversity loss in RLVR. Yue et al. (Yue et al., 2025) show RLVR narrows reasoning boundaries. Negative sample reinforcement (Zhu et al., 2025) demonstrates the importance of learning from incorrect rollouts but does not differentiate among error types. We identify overconfident errors as a key mechanism driving diversity collapse and propose confidence-based differential penalization to address this.

3 Preliminaries

Setting. We consider a policy π_θ parameterized by θ, initialized from a reference model π_{ref}. Given a prompt $x \sim \mathcal{D}$, the model generates G rollouts $\{y_1, \ldots, y_G\}$. Each rollout receives a reward $r_i \in \mathbb{R}$ from a reward function or verifier. We define:

- Empirical mean reward: $\hat{\mu}_x = \frac{1}{G}\sum_{i=1}^{G} r_i$
- Empirical reward standard deviation: $\hat{\sigma}_x = \sqrt{\frac{1}{G}\sum_{i=1}^{G}(r_i - \hat{\mu}_x)^2}$

- Above-average rollout set: $\mathcal{Y}^+(x) = \{y_i : r_i > \hat{\mu}_x\}$
- Below-average rollout set: $\mathcal{Y}^-(x) = \{y_i : r_i \leq \hat{\mu}_x\}$

GRPO objective. In Group Relative Policy Optimization (Shao et al., 2024), the advantage for rollout y_i is computed via group normalization:

$$\hat{A}_i = \frac{r_i - \hat{\mu}_x}{\hat{\sigma}_x + \epsilon} \tag{1}$$

where ϵ is a small constant for numerical stability. The clipped surrogate objective is:

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\mathbb{E}_{x\sim\mathcal{D}}\left[\frac{1}{G}\sum_{i=1}^{G}\min\Big(\rho_i\hat{A}_i,\ \text{clip}(\rho_i, 1{-}\epsilon_c, 1{+}\epsilon_c)\hat{A}_i\Big)\right] + \beta\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}}) \tag{2}$$

where $\rho_i = \pi_\theta(y_i|x)/\pi_{\text{old}}(y_i|x)$ is the importance sampling ratio and ϵ_c is the clipping threshold.

Observation: uniform penalty within groups. For rollouts with identical rewards $r_i = r_j$, the advantages are also identical: $\hat{A}_i = \hat{A}_j$. In the special case of binary rewards, all incorrect rollouts ($r_i = 0$) share the same advantage:

$$\hat{A}_i^- = \frac{-\hat{\mu}_x}{\hat{\sigma}_x + \epsilon} \tag{3}$$

More generally, rollouts with the same reward receive *identical* advantage values regardless of their qualitative differences. The only per-rollout modulation comes from the importance ratio ρ_i, which is bounded by clipping and provides limited differentiation.

Motivation: overconfident errors. Define the per-rollout *confidence shift* $c_i = \log(\pi_\theta(y_i|x)/\pi_{\text{ref}}(y_i|x))$: positive values indicate the policy has become more confident than the reference on rollout y_i, while negative values indicate the opposite. Training Qwen2.5-Math-7B with standard GRPO on DAPO-Math-17K (Yu et al., 2025), we observe that the distribution of c_i among *incorrect* rollouts develops a heavy right tail as training progresses—a substantial fraction of errors become significantly more probable under the trained policy than under the reference, even though they remain incorrect. This is consistent with analyses of implicit reward distributions in preference optimization (Rafailov et al., 2023b; Meng et al., 2024). These overconfident errors consume probability mass that would otherwise support diverse reasoning paths, contributing to the diversity collapse documented by Yue et al. (2025). Crucially, the standard global KL penalty $\beta\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}})$ cannot selectively address this: it penalizes *all* deviations from the reference proportionally, suppressing beneficial confidence growth on correct paths alongside harmful overconfidence on incorrect ones. This structural limitation motivates a *targeted* correction mechanism (see §5.3 for detailed quantitative tracking).

4 The ACE Method

4.1 Error Confidence Score

Definition 1 (Error Confidence Score)**.** *For a prompt x and an incorrect rollout $y_i \in \mathcal{Y}^-(x)$, the **error confidence score** is:*

$$c_i \triangleq \log\frac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)} = \sum_{t=1}^{T_i}\log\frac{\pi_\theta(y_i^{(t)}|x, y_i^{(<t)})}{\pi_{\text{ref}}(y_i^{(t)}|x, y_i^{(<t)})} \tag{4}$$

where $y_i^{(t)}$ denotes the t-th token and T_i is the sequence length.

The second equality decomposes the sequence-level confidence into a sum of *token-level* log-ratios. This is important for two reasons: (a) it shows that c_i is already computed as a byproduct of standard RLVR training (which requires $\log\pi_\theta$ and $\log\pi_{\text{ref}}$ for the KL penalty and importance ratios), incurring *zero additional compute*; and (b) it reveals that c_i aggregates confidence shifts across all reasoning steps, naturally weighting tokens where the policy has diverged most from the reference.

Remark 1 (Three regimes)**.** *The sign of c_i partitions incorrect rollouts into interpretable regimes:*

- $c_i > 0$*:* ***Overconfident errors.*** *The policy assigns* higher *probability than the reference. These are spurious patterns actively learned during RL.*
- $c_i \approx 0$*:* ***Exploratory errors.*** *Probability approximately unchanged from the reference. Natural stochastic deviations.*
- $c_i < 0$*:* ***Self-correcting errors.*** *The policy has already reduced probability mass relative to the reference.*

4.2 The ACE Advantage

We restructure the negative advantage to depend on the per-rollout confidence score c_i.

Definition 2 (ACE Advantage)**.** *For an incorrect rollout $y_i \in \mathcal{Y}^-(x)$, the ACE advantage is:*

$$A^-_{\text{ACE},i} = \hat{A}^-_i \cdot (1 + \alpha \cdot \text{Softplus}(c_i)) \tag{5}$$

where $\hat{A}^-_i = (r_i - \hat{\mu}_x)/(\hat{\sigma}_x + \epsilon)$ is the standard GRPO advantage for incorrect rollouts and $\alpha \geq 0$ is a hyperparameter controlling the correction strength. Since $\hat{A}^-_i < 0$ and $(1 + \alpha \cdot \text{Softplus}(c_i)) \geq 1$, ACE strictly amplifies the magnitude of the penalty. For correct rollouts $y_i \in \mathcal{Y}^+(x)$, we retain the standard GRPO advantage:

$$A^+_{\text{ACE},i} = \hat{A}_i = \frac{r_i - \hat{\mu}_x}{\hat{\sigma}_x + \epsilon} \tag{6}$$

Design rationale. The Softplus function $\text{Softplus}(z) = \log(1 + e^z)$ is chosen for three properties:

1. **Asymptotic behavior.** When $c_i \gg 0$ (overconfident), $\text{Softplus}(c_i) \approx c_i$: penalty scales linearly with the log-confidence ratio. When $c_i \ll 0$ (self-correcting), $\text{Softplus}(c_i) \approx e^{c_i} \to 0$: penalty converges to the base GRPO advantage $\hat{A}^-_i$.
2. **Smoothness.** Unlike $\max(0, c_i)$ (which has a non-differentiable kink at 0), Softplus is infinitely differentiable everywhere, ensuring smooth gradient flow.
3. **Monotonicity.** Softplus is strictly increasing, so more confident errors always receive strictly larger penalties, consistent with our theoretical motivation.

Comparison to uniform penalization. To illustrate the effect of ACE, consider the binary reward case where rollouts receive $r_i \in \{0, 1\}$. Under standard GRPO, all incorrect rollouts ($r_i = 0$) share the same advantage $\hat{A}^- = -\hat{p}_x/(\hat{\sigma}_x + \epsilon)$, where $\hat{p}_x$ is the empirical pass rate and $\hat{\sigma}_x = \sqrt{\hat{p}_x(1 - \hat{p}_x)}$.

Difficulty-adaptive scaling. Since ACE multiplies the standard GRPO advantage $\hat{A}^-_i$ by $(1 + \alpha \cdot \text{Softplus}(c_i))$, it naturally inherits GRPO's difficulty-dependent scaling: easy prompts (high pass rate) produce larger $|\hat{A}^-_i|$, so errors on easy problems are penalized more heavily. The confidence modulation then provides *additional* per-rollout differentiation within each difficulty level.

Penalty differentiation. Under ACE:

$$A^-_{\text{ACE},i} = \hat{A}^-_i \cdot (1 + \alpha \log(1 + e^{c_i})) \quad \Longrightarrow \quad |A^-_{\text{ACE},i}| \text{ is strictly increasing in } c_i \tag{7}$$

Therefore, within the same group, an overconfident error ($c_i = 2$) receives a penalty $|\hat{A}^-| \cdot (1 + \alpha \cdot 2.13)$ while an exploratory error ($c_i = 0$) receives $|\hat{A}^-| \cdot (1 + \alpha \cdot 0.69)$, and a self-correcting error ($c_i = -3$) receives $|\hat{A}^-| \cdot (1 + \alpha \cdot 0.05)$. This provides fine-grained differentiation that is impossible under uniform penalization. The same principle extends to continuous rewards, where ACE differentiates among below-average rollouts based on their confidence scores.

ACE in One Sentence

Standard RLVR punishes all wrong answers equally. ACE punishes wrong answers *the model has learned to be confident in* much harder, while leaving natural exploration mistakes alone.

Algorithm 1 ACE-GRPO: Asymmetric Confidence-aware Error Penalty

Require: Policy π_θ, reference model π_{ref}, prompt dataset $\mathcal{D}$, group size G, ACE strength α, clipping ϵ_c, KL coefficient β

for each training step **do**
  Sample prompt batch $\{x_1, \ldots, x_B\} \sim \mathcal{D}$
  for each prompt x in batch **do**
    Generate G rollouts $\{y_1, \ldots, y_G\} \sim \pi_\theta(\cdot|x)$
    Compute rewards $r_i \in \{0, 1\}$ via verifier
    Compute standard group advantages $\hat{A}_i$ via GRPO
    for each incorrect rollout y_i with $r_i = 0$ **do**
      $c_i \leftarrow \left(\sum_{t=1}^{T_i} \log \pi_\theta(y_i^{(t)}|\cdot) - \log \pi_{\text{ref}}(y_i^{(t)}|\cdot)\right)/T_i$ // Already computed
      $A^-_{\text{ACE},i} \leftarrow \hat{A}_i^- \cdot (1 + \alpha \cdot \log(1 + \exp(c_i)))$ // Amplify uniform advantage
    end for
    Compute clipped surrogate loss (Eq. 8)
  end for
  Update θ via gradient descent
end for

4.3 ACE-GRPO: Integration and Algorithm

ACE modifies only the advantage computation for negative samples. Substituting the ACE advantage (Definition 2) into the GRPO objective (Eq. 2), the full ACE-GRPO objective is:

$$\mathcal{L}_{\text{ACE}}(\theta) = -\mathbb{E}_{x\sim\mathcal{D}}\left[\frac{1}{G}\sum_{i=1}^{G}\left(\mathbb{I}[r_i{=}1]\cdot\mathcal{L}_i^+ + \mathbb{I}[r_i{=}0]\cdot\mathcal{L}_i^-\right)\right] + \beta\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}}) \tag{8}$$

where:

$$\mathcal{L}_i^+ = \min\Big(\rho_i\hat{A}_i^+,\ \text{clip}(\rho_i, 1{-}\epsilon_c, 1{+}\epsilon_c)\hat{A}_i^+\Big) \tag{9}$$

$$\mathcal{L}_i^- = \min\Big(\rho_i A^-_{\text{ACE},i},\ \text{clip}(\rho_i, 1{-}\epsilon_c, 1{+}\epsilon_c)A^-_{\text{ACE},i}\Big) \tag{10}$$

The positive advantages $\hat{A}_i^+$ retain the standard GRPO formulation.

Practical considerations. In practice, we normalize c_i by sequence length ($\bar{c}_i = c_i/T_i$) to ensure comparable penalty magnitudes across rollouts of different lengths. Additional implementation details (sequence-level vs. token-level aggregation, clipping choices) and a PyTorch implementation are provided in Appendix C. The full algorithm is given below.

4.4 Relationship to Selective Reverse KL Divergence

We now characterize the theoretical relationship between ACE's additional penalty and a selective regularizer that targets overconfident errors. Crucially, the exact decomposition holds for the *unclipped on-policy policy-gradient term* underlying ACE-GRPO, with the group advantages treated as fixed scalars via stop-gradient, as is standard in actor updates. The practical clipped GRPO objective should therefore be viewed as a local approximation to this base case: the result is exact when the importance ratio is evaluated at $\rho_i = 1$ (equivalently, $\pi_{\text{old}} = \pi_\theta$) or when clipping is inactive, and otherwise serves as the first-order policy-gradient interpretation of ACE.

Theorem 1 (Selective Regularization Decomposition)**.** *Let $\widetilde{\mathcal{L}}_{\text{std}}(\theta)$ and $\widetilde{\mathcal{L}}_{\text{ACE}}(\theta)$ denote the* unclipped on-policy actor objectives *obtained from Eq. 2 and Eq. 8 by evaluating the importance ratio at $\rho_i = 1$ (equivalently, $\pi_{\text{old}} = \pi_\theta$, or the local regime where clipping is inactive) and by treating the group-level quantities $\hat{\mu}_x$, $\hat{\sigma}_x$, and $\hat{A}^-(x)$ as fixed with respect to θ (stop-gradient through the advantage computation). Define the* ***selective***

***regularizer**:*

$$\mathcal{R}_{\text{sel}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\left[|\hat{A}^{-}(x)| \sum_{y\in\mathcal{Y}^{-}(x)} \pi_\theta(y|x)\cdot \text{Softplus}\left(\log\frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}\right)\right] \tag{11}$$

where $|\hat{A}^{-}(x)|$ is the magnitude of the standard GRPO negative advantage for prompt x. Assume rollouts are sampled on-policy from π_θ. Then, in the infinite-sample limit ($G\to\infty$), the α-dependent additional gradient from ACE decomposes exactly *as:*

$$\Delta\nabla_\theta \;=\; -\alpha\nabla_\theta\mathcal{R}_{\text{sel}}(\theta) \;+\; \alpha\,\mathbb{E}_{x\sim\mathcal{D}}\left[|\hat{A}^{-}(x)| \sum_{y\in\mathcal{Y}^{-}(x)} \pi_\theta(y|x)\,\sigma(c(y))\,\nabla_\theta\log\pi_\theta(y|x)\right] \tag{12}$$

where $\sigma(c) = 1/(1+e^{-c})$ is the sigmoid function (i.e., $\text{Softplus}'(c)$*). Equivalently, ACE implements the* negative *gradient of $\mathcal{R}_{\text{sel}}$ with the confidence modulation treated as a* fixed reward signal *(stop-gradient on c_i), plus the residual term $\mathcal{E}(\theta)$:*

$$\mathcal{E}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\left[|\hat{A}^{-}(x)| \sum_{y\in\mathcal{Y}^{-}(x)} \pi_\theta(y|x)\,\sigma(c(y))\,\nabla_\theta\log\pi_\theta(y|x)\right] \tag{13}$$

Moreover, for overconfident errors where $c(y)\gg 0$, $\text{Softplus}(c)\approx c$*, and the dominant term in $\mathcal{R}_{\text{sel}}$ takes the form of a* difficulty-weighted *reverse KL divergence restricted to overconfident incorrect trajectories:*

$$\mathcal{R}_{\text{sel}}(\theta) \;\approx\; \mathbb{E}_{x\sim\mathcal{D}}\left[|\hat{A}^{-}(x)| \sum_{\substack{y\in\mathcal{Y}^{-}(x)\\ c(y)>0}} \pi_\theta(y|x)\cdot\log\frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}\right] \tag{14}$$

The proof is provided in Appendix A. Intuitively, ACE's stop-gradient treatment of $\text{Softplus}(c_i)$ captures the dominant selective-regularization component (Term I: confidence-weighted probability suppression), while the residual $\mathcal{E}(\theta)$ corresponds to the through-c_i gradient (Term II) that the full regularizer would additionally apply. By omitting Term II, ACE implements a *tempered* version of $\mathcal{R}_{\text{sel}}$—less aggressive than the full regularizer, but more targeted than standard GRPO. The per-prompt factor $|\hat{A}^{-}(x)|$ ensures that the selective regularizer inherits the difficulty-adaptive scaling of GRPO. In contrast to the global KL term $\beta\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}})$ which indiscriminately pulls back *all* deviations, $\mathcal{R}_{\text{sel}}$ is (i) restricted to incorrect outputs ($y\in\mathcal{Y}^{-}$), (ii) activated primarily by overconfidence ($c_i>0$) due to Softplus saturation, (iii) independently tunable via α, and (iv) difficulty-adaptive via the $|\hat{A}^{-}(x)|$ factor.

4.5 Gradient Quality Analysis

A natural question is whether ACE's confidence-dependent reweighting improves or degrades gradient quality. We analyze this in detail in Appendix B and summarize the key results here.

First, ACE necessarily increases both the total gradient second moment and the directional variance—unavoidable consequences of additive reweighting where $(1+\alpha\phi_i)>1$ for all $\phi_i>0$ (Proposition 1). However, this does *not* prevent quality improvement. We define the gradient quality ratio as $Q_d = \mu_d^2/\sigma_d^2$, measuring the ratio of squared directional signal to directional variance. Under realistic conditions—specifically, when overconfident errors carry gradients aligned with the optimization direction ($\text{Cov}(\phi_i,u_i)>0$) and the baseline gradient is noisy ($Q_d^{\text{std}}<1$)—we prove that ACE strictly improves gradient quality: $Q_d^{\text{ACE}} > Q_d^{\text{std}}$ (Theorem 2). The key mechanism is that ACE's selective amplification concentrates extra weight on the most informative gradients, causing the signal to grow faster than the noise along the optimization-relevant direction.

Theoretical Insight: Gradient Efficiency

ACE converts *harmful variance into exploitable signal*. By concentrating penalty weight on overconfident errors—which have gradients more aligned with the optimization direction—ACE achieves higher signal-to-noise ratio despite increasing total gradient variance. This is the statistical foundation for ACE's improved learning efficiency.

5 Experiments: ACE Expands the Reasoning Boundary

5.1 Experimental Setup

Models. We fine-tune Qwen2.5-Math-7B (Qwen Team et al., 2024), Qwen3-8B-Base (Yang et al., 2025), and Llama-3.1-8B-Instruct (Grattafiori et al., 2024) using GRPO implemented with VERL (Volcano Engine, 2024). Note that Qwen3-8B-Base is evaluated *without* enabling the extended thinking mode (i.e., reasoning mode disabled). Llama-3.1-8B-Instruct is included in the main results (Tables 1 and 2) to test cross-family generalization beyond the Qwen model family. For the detailed diagnostic experiments (§5.3–§5.4), ablations (§5.5), and hyperparameter sensitivity (Appendix D), we focus on the two Qwen models because: (i) they serve as the primary experimental subjects and already span two distinct pretraining recipes (math-specialized vs. general-purpose base model), providing sufficient diversity to validate the generality of our findings; and (ii) Llama-3.1-8B-Instruct operates in a substantially lower accuracy regime (e.g., near-floor on AIME 2025), which makes fine-grained diagnostics such as overconfident error distributions and entropy dynamics less statistically informative.

Training data. We use the DAPO-Math-17K dataset (Yu et al., 2025) as the training prompts. For the GRPO and ACE-GRPO baselines we use standard GRPO (symmetric clipping, with KL penalty); for DAPO and ACE-DAPO we use the full DAPO algorithm (Yu et al., 2025) with Clip-Higher, dynamic sampling, and token-level loss.

Evaluation. We evaluate on MATH-500 (Hendrycks et al., 2021) and AIME 2025 using a rule-based math verifier for correctness verification.

Metrics. We report Pass@k for $k \in \{1, 2, 4, 8, 16, 32\}$ using temperature 0.7 and top-$p = 0.95$. Pass@k measures the probability that at least one of k samples is correct. We use the unbiased estimator from Chen et al. (2021):

$$\text{Pass@}k = \mathbb{E}_{x\sim\mathcal{D}}\left[1 - \frac{\binom{n-c}{k}}{\binom{n}{k}}\right] \tag{15}$$

where n is the total samples and c is the number correct. Pass@1 reflects exploitation; large-k reflects exploration and reasoning boundary.

Baselines.

- **Base model**: Unmodified pretrained model (upper bound for large-k diversity).
- **GRPO**: Standard Group Relative Policy Optimization (Shao et al., 2024).
- **DAPO**: The full DAPO algorithm (Yu et al., 2025), which uses asymmetric clipping (Clip-Higher), dynamic sampling, and token-level loss, trained on the same DAPO-Math-17K dataset.
- **ACE-GRPO**: Our method (ACE applied to GRPO).
- **ACE-DAPO**: Our method applied on top of DAPO, demonstrating composability with orthogonal diversity-preserving strategies.

Hyperparameters. For ACE, we set $\alpha = 1.0$ as the default. We use normalized confidence scores $\bar{c}_i = c_i/T_i$. Full training hyperparameters are provided in Appendix E.

Table 1: Pass@k (%) on MATH-500. We report mean ± 95% confidence interval over 5 independent training runs. **Bold** = best within each model group.

Model	@1	@2	@4	@8	@16	@32
Qwen2.5-Math-7B	63.0	76.3	83.2	88.1	91.2	93.5
Qwen2.5-Math-7B + GRPO	73.4 ± 0.8	79.5 ± 0.7	83.2 ± 0.7	86.2 ± 0.5	89.7 ± 0.5	91.3 ± 0.5
Qwen2.5-Math-7B + DAPO	74.5 ± 1.0	80.8 ± 0.9	84.8 ± 0.8	89.5 ± 0.7	93.0 ± 0.7	94.6 ± 0.6
Qwen2.5-Math-7B + ACE-GRPO	74.2 ± 0.7	80.9 ± 0.7	84.5 ± 0.6	88.9 ± 0.5	92.6 ± 0.5	94.3 ± 0.4
Qwen2.5-Math-7B + ACE-DAPO	**75.1** ± 0.8	**82.4** ± 0.8	**86.2** ± 0.6	**91.2** ± 0.5	**94.7** ± 0.5	**96.1** ± 0.5
Qwen3-8B-Base	60.2	72.5	78.8	83.9	87.2	90.6
Qwen3-8B-Base + GRPO	69.4 ± 0.9	75.5 ± 0.9	79.3 ± 0.9	82.5 ± 0.9	86.2 ± 0.8	88.6 ± 0.7
Qwen3-8B-Base + DAPO	70.8 ± 1.0	76.8 ± 0.9	81.1 ± 0.9	84.3 ± 0.8	88.1 ± 0.8	90.4 ± 0.8
Qwen3-8B-Base + ACE-GRPO	70.1 ± 0.7	76.5 ± 0.7	81.1 ± 0.6	84.5 ± 0.6	88.5 ± 0.6	91.1 ± 0.5
Qwen3-8B-Base + ACE-DAPO	**71.2** ± 0.9	**77.5** ± 0.9	**82.3** ± 0.8	**85.4** ± 0.8	**89.4** ± 0.7	**91.6** ± 0.7
Llama-3.1-8B-Instruct	48.1	59.9	67.8	**74.8**	**80.5**	**84.8**
Llama-3.1-8B-Instruct + GRPO	52.9 ± 1.1	60.5 ± 1.0	67.3 ± 0.9	71.8 ± 0.9	75.5 ± 0.9	79.3 ± 0.8
Llama-3.1-8B-Instruct + DAPO	54.3 ± 1.0	61.8 ± 1.0	68.9 ± 1.0	72.9 ± 1.0	76.8 ± 0.9	80.4 ± 0.9
Llama-3.1-8B-Instruct + ACE-GRPO	54.1 ± 1.1	62.2 ± 1.1	69.1 ± 1.0	73.5 ± 1.0	76.8 ± 0.9	81.5 ± 0.9
Llama-3.1-8B-Instruct + ACE-DAPO	**55.4** ± 1.1	**62.8** ± 1.1	**70.2** ± 1.0	74.1 ± 1.0	77.9 ± 0.9	82.1 ± 0.9

Table 2: Pass@k (%) on AIME 2025. AIME 2025 contains 30 problems; we report point estimates as confidence intervals are dominated by test-set size rather than training variance. **Bold** = best within each model group.

Model	@1	@2	@4	@8	@16	@32
Qwen2.5-Math-7B	6.3	9.9	13.8	17.5	21.9	26.7
Qwen2.5-Math-7B + GRPO	10.5	14.9	19.7	23.9	28.6	33.7
Qwen2.5-Math-7B + DAPO	11.5	16.7	22.5	27.5	31.8	37.1
Qwen2.5-Math-7B + ACE-GRPO	11.2	16.0	21.2	26.1	30.6	36.4
Qwen2.5-Math-7B + ACE-DAPO	**11.7**	**17.4**	**23.8**	**28.5**	**33.1**	**38.6**
Qwen3-8B-Base	5.1	9.2	11.6	14.2	17.0	19.6
Qwen3-8B-Base + GRPO	9.7	13.9	17.4	22.5	25.7	29.8
Qwen3-8B-Base + DAPO	11.1	15.7	19.9	25.2	28.5	33.1
Qwen3-8B-Base + ACE-GRPO	10.5	15.5	19.6	24.7	27.9	32.4
Qwen3-8B-Base + ACE-DAPO	**11.2**	**16.9**	**21.2**	**26.3**	**29.9**	**34.4**
Llama-3.1-8B-Instruct	0.2	**0.7**	**1.2**	**3.2**	**7.1**	**10.8**
Llama-3.1-8B-Instruct + GRPO	0.2	0.3	0.5	2.1	3.0	7.0
Llama-3.1-8B-Instruct + DAPO	**0.3**	0.3	0.6	1.9	2.8	6.3
Llama-3.1-8B-Instruct + ACE-GRPO	**0.3**	0.3	0.5	2.2	3.9	8.2
Llama-3.1-8B-Instruct + ACE-DAPO	0.2	0.3	0.6	2.0	3.2	7.1

Fair comparison. Within each model, we keep the training recipe and budget matched across methods; see Appendix E.

5.2 Main Results: Full Pass@k Spectrum

Table 1 reports Pass@k on MATH-500 and Table 2 reports results on AIME 2025.

Key findings (Qwen2.5-Math-7B). As shown in Figure 2, ACE consistently improves larger-k metrics while maintaining comparable Pass@1. On MATH-500, ACE-GRPO improves Pass@32 from 91.3% to 94.3% (+3.0pp) over GRPO; ACE-DAPO further pushes Pass@32 to 96.1% (+1.5pp over DAPO's 94.6%). On

AIME 2025, ACE-GRPO improves Pass@32 from 33.7% to 36.4% (+2.7pp); ACE-DAPO reaches 38.6% (+1.5pp over DAPO's 37.1%). Notably, ACE-DAPO achieves the strongest results across all k, demonstrating that ACE composes effectively with orthogonal diversity-preserving strategies.

Key findings (Qwen3-8B-Base). The same pattern holds on a different model family. On MATH-500, ACE-GRPO improves Pass@32 from 88.6% to 91.1% (+2.5pp); ACE-DAPO reaches 91.6% (+1.2pp over DAPO's 90.4%). On AIME 2025, ACE-GRPO improves Pass@32 from 29.8% to 32.4% (+2.6pp); ACE-DAPO reaches 34.4% (+1.3pp over DAPO's 33.1%).

Key findings (Llama-3.1-8B-Instruct). To test cross-family generalization, we evaluate on a non-Qwen model. On MATH-500, ACE-GRPO improves Pass@32 from 79.3% to 81.5% (+2.2pp); ACE-DAPO reaches 82.1% (+1.7pp over DAPO's 80.4%). On AIME 2025—where Llama-3.1-8B-Instruct operates near the floor—ACE-GRPO improves Pass@32 from 7.0% to 8.2% (+1.2pp), while ACE-DAPO does not deliver a consistent gain over DAPO or the base model. We therefore view the Llama results as directional evidence that ACE's mechanism can transfer across model families, with the clearest support coming from the stronger MATH-500 gains and the ACE-GRPO improvement at large k on AIME 2025.

Taken together, the two Qwen families provide the strongest evidence for ACE's generality, while the Llama results offer additional but weaker support in a lower-accuracy regime.

Interaction with DAPO's Clip-Higher. A natural observation is that ACE's marginal gain over DAPO is smaller than over GRPO (e.g., on MATH-500 Qwen2.5-Math-7B Pass@32: +3.0pp for ACE-GRPO vs. GRPO, but +1.5pp for ACE-DAPO vs. DAPO). This reflects a genuine mechanism overlap: DAPO's Clip-Higher preserves diversity by limiting how aggressively *any* incorrect path is suppressed at the token level, which indirectly reduces the overconfident-error pathology that ACE targets. However, DAPO's protection is *indiscriminate*—it shields overconfident errors and exploratory errors alike, because token-level clipping cannot distinguish trajectory-level confidence regimes. ACE provides the missing selectivity: it amplifies suppression specifically for errors the model has learned to be confident in, while leaving exploratory errors untouched. The consistent gains of ACE-DAPO over DAPO across all model families and benchmarks indicate that this trajectory-level selectivity captures a dimension of the overconfidence problem that token-level clipping alone cannot resolve. The diminishing marginal returns are expected—both methods partially address the same pathology—but the residual improvement confirms that ACE's rollout-level discrimination provides value beyond what DAPO's uniform token-level mechanism achieves.

Main Result

ACE preserves Pass@1 performance while significantly expanding the reasoning boundary at large k. On the two Qwen families, ACE-GRPO consistently improves Pass@32 by +2.5–3.0pp over GRPO, and ACE-DAPO further improves over DAPO by +1.2–1.5pp, showing that ACE provides complementary correction beyond token-level diversity strategies. On Llama-3.1-8B-Instruct, ACE-GRPO also improves Pass@32, while ACE-DAPO yields mixed results in the low-accuracy AIME 2025 regime.

5.3 Experiment 1: Overconfident Error Dynamics

Goal. Quantify the prevalence of overconfident errors during training and demonstrate that ACE effectively reduces them.

Design. Track the distribution of c_i among incorrect rollouts throughout training for both standard GRPO and ACE-GRPO on the two Qwen models.[1] At checkpoints every 25 training steps, generate 32 rollouts per prompt on a held-out set and record c_i for all incorrect rollouts.

[1]We omit Llama-3.1-8B-Instruct from the diagnostic experiments as its lower baseline accuracy yields fewer correct rollouts per group, making the confidence shift statistics noisier and less informative. The main results in Tables 1–2 confirm that ACE's gains transfer to Llama.

Metrics.

- **Overconfident error fraction**: $\mathrm{OEF}(t) = |\{y_i \in \mathcal{Y}^- : c_i > 0\}|/|\mathcal{Y}^-|$ at step t.
- **Mean overconfidence magnitude**: $\mathbb{E}[c_i \mid c_i > 0, r_i = 0]$ at step t.
- **Token-level entropy**: Average per-token entropy of the policy.

Results. The core claim of ACE is that standard GRPO allows incorrect rollouts to become increasingly overconfident during training, and that ACE's asymmetric penalty should counteract this pathology. To test this, we track two complementary diagnostics at every checkpoint (Figure 3): (i) the *overconfident error fraction* (OEF), which measures the proportion of incorrect rollouts whose confidence has grown relative to the reference policy ($c_i > 0$), and (ii) the *mean overconfidence magnitude* among those overconfident errors, which captures the severity of the problem. Throughout training, ACE-GRPO maintains a lower OEF and a lower mean overconfidence magnitude than standard GRPO at every recorded checkpoint, indicating that ACE consistently suppresses both the prevalence and the severity of high-confidence incorrect rollouts.

5.4 Experiment 2: Entropy Dynamics

Goal. Verify that ACE preserves generation diversity by tracking entropy throughout training, and establish the connection between entropy and Pass@k performance.

Design. Over the first 20 training steps, compute the average per-token entropy of the policy on a held-out subset of DAPO-Math-17K prompts:

$$H(t) = -\frac{1}{|\mathcal{D}_{\text{val}}|} \sum_{x \in \mathcal{D}_{\text{val}}} \frac{1}{T} \sum_{j=1}^{T} \sum_{v} \pi_\theta(v|x, y_{<j}) \log \pi_\theta(v|x, y_{<j}) \tag{16}$$

where T is the average sequence length and v ranges over the vocabulary.

Results. A key concern with aggressive error suppression is that it may cause premature mode collapse, concentrating probability mass on a narrow set of outputs and destroying the diversity needed for high Pass@k at large k. To diagnose this, we track average per-token entropy $H(t)$ over the early phase of training, where entropy decay is most rapid, for both standard GRPO and ACE-GRPO (Figure 4). Standard GRPO exhibits a sharp entropy drop within the first 20 steps, retaining only a small fraction of its initial entropy. In contrast, ACE-GRPO decays substantially more slowly, preserving a much larger fraction of the initial entropy over the same period. This gap correlates with Pass@k performance at large k: the method that retains more entropy also achieves higher coverage, confirming that ACE's selective penalty avoids premature mode collapse while still suppressing overconfident errors.

5.5 Ablation: Choice of Modulation Function

A natural question is whether the choice of Softplus as the modulation function is important, or whether a simpler alternative such as $\mathrm{ReLU}(c_i) = \max(0, c_i)$ suffices. We compare the two variants on MATH-500 using Qwen2.5-Math-7B with $\alpha = 1.0$ (the ablation uses a single representative model to isolate the effect of the modulation function; the main results in Table 1 confirm that ACE's gains are consistent across all three model families):

- **ACE-Softplus** (default): $A^-_{\text{ACE},i} = \hat{A}^-_i \cdot (1 + \alpha \cdot \mathrm{Softplus}(c_i))$
- **ACE-ReLU**: $A^-_{\text{ACE},i} = \hat{A}^-_i \cdot (1 + \alpha \cdot \mathrm{ReLU}(c_i))$

ReLU completely ignores self-correcting and exploratory errors ($c_i \leq 0$), providing zero modulation in that regime, while Softplus provides a smooth, everywhere-positive modulation that transitions gradually.

Table 3: Ablation: modulation function on MATH-500 (Qwen2.5-Math-7B, $\alpha = 1.0$).

Method	**@1**	**@2**	**@4**	**@8**	**@16**	**@32**
GRPO (baseline)	73.4	79.5	83.2	86.2	89.7	91.3
ACE-ReLU	73.2	80.3	83.9	87.6	91.2	93.1
ACE-Softplus (ours)	74.2	80.9	84.5	88.9	92.6	94.3

Analysis. Both ACE-ReLU and ACE-Softplus outperform standard GRPO across all $k > 1$, confirming that confidence-aware modulation—regardless of the specific activation—is beneficial. However, ACE-Softplus consistently outperforms ACE-ReLU, with the gap widening at larger k (+1.2 pp at Pass@32). This advantage stems from two properties of Softplus. First, *smoothness*: ReLU has a non-differentiable kink at $c_i = 0$, creating a discontinuity in the gradient landscape that can destabilize training, whereas Softplus provides smooth gradient flow everywhere. Second, *non-zero modulation near the boundary*: ReLU assigns zero modulation to all errors with $c_i \leq 0$, treating them identically to standard GRPO. In contrast, Softplus(0) $= \ln 2 \approx 0.69$, providing a gentle baseline modulation that enables finer differentiation among borderline errors near $c_i \approx 0$—precisely the regime where errors may be transitioning from exploratory to overconfident. These results empirically validate the design rationale in §4.2.

5.6 Analysis: Mechanism Behind Diversity Preservation

The experimental results above (§5.3–§5.4) reveal a consistent mechanism: standard GRPO's uniform penalties allow overconfident errors to form "probability sinks" that crowd out valid reasoning paths—the pathology identified by Yue et al. (2025) as the root cause of RLVR's narrowing reasoning boundary. ACE's asymmetric penalties break this cycle: the selective KL term (Theorem 1) acts as entropy regularization restricted to the overconfident region, while leaving exploratory errors ($c_i \leq 0$) untouched. This targeted correction redistributes probability mass to alternative reasoning paths, explaining ACE's improvements across the full Pass@k spectrum.

6 Limitations and Future Work

Dependence on reference model quality. ACE uses π_{ref} to define overconfidence. If the reference model is poorly calibrated, the confidence score c_i may not reliably indicate spurious patterns. Exploring alternatives (e.g., using a moving average of recent checkpoints) is a direction for future work.

Binary rewards only. Our current formulation assumes binary rewards ($r \in \{0, 1\}$). Extending ACE to continuous or partial rewards (e.g., from process reward models) requires redefining what constitutes an "overconfident error" in the presence of graded feedback.

Interaction with long CoT. Extended reasoning models (e.g., with >10K token outputs) may exhibit different confidence shift dynamics. The sequence-length normalization ($\bar{c}_i = c_i/T_i$) may need refinement for very long chains.

7 Conclusion

We identified a previously overlooked pathology in RLVR training: the accumulation of overconfident errors—incorrect reasoning paths that the RL process spuriously reinforces. We proposed ACE, a simple modification to the advantage function that dynamically amplifies penalties for overconfident errors while leaving exploratory errors untouched.

Core Contributions

(1) We formalize *error confidence shift* as a new per-rollout diagnostic orthogonal to prompt difficulty, revealing that overconfident errors accumulate during RLVR and drive diversity collapse. **(2)** ACE's gradient decomposes into a *selective regularizer* on overconfident errors plus a tempering residual, providing principled theoretical grounding. **(3)** ACE improves the full Pass@k spectrum—especially at large k—without sacrificing Pass@1, adding only a single Softplus computation per incorrect rollout.

Reproducibility Statement

We provide full implementation details to facilitate reproducibility. The ACE algorithm modifies only the advantage computation (Definition 2), requiring a single Softplus operation per incorrect rollout with zero additional compute beyond the standard RLVR pipeline. A complete PyTorch implementation is given in Appendix C—the core change is fewer than 10 lines of code on top of any standard GRPO/DAPO training loop. All training hyperparameters are reported in Appendix E (Table 5), including learning rates, batch sizes, rollout group sizes, and clipping/KL coefficients for all three model families. Within each model, we use the same training recipe across methods to ensure fair comparison. We train with VERL (Volcano Engine, 2024) on the publicly available DAPO-Math-17K dataset (Yu et al., 2025) and evaluate on the publicly available MATH-500 (Hendrycks et al., 2021) and AIME 2025 benchmarks. For MATH-500, we report mean $\pm$ 95% confidence intervals over 5 independent training runs. All proofs are provided in the appendix with full derivations. While we are unable to release our training codebase due to organizational constraints, ACE does not rely on proprietary infrastructure beyond a standard VERL-based RLVR pipeline. The simplicity of ACE (a single-function modification to the advantage computation), together with the pseudocode in Algorithm 1 and the PyTorch snippet in Appendix C, should enable straightforward reimplementation.

References

B. Brown, J. Juravsky, R. Ehrlich, R. Clark, Q. V. Le, C. Ré, and A. Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

M. Chen, J. Tworek, H. Jun, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

DeepSeek-AI, D. Guo, D. Yang, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. 2026. doi: https://doi.org/10.1038/s41586-025-09422-z. URL `https://arxiv.org/abs/2501.12948`.

S. Devlin and D. Kudenko. Dynamic potential-based reward shaping. In *AAMAS*, pp. 433–440, 2012.

A. Grattafiori, A. Dubey, A. Jauhri, et al. The llama 3 herd of models. 2024. URL `https://arxiv.org/abs/2407.21783`.

D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. Measuring mathematical problem solving with the math dataset. In *NeurIPS*, 2021.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=v8L0pN6EOi`.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=3Tzcot1LKb`.

A. Y. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, pp. 278–287, 1999.

OpenAI, A. Jaech, A. Kalai, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, and R. Lowe. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

S. Parashar, S. Gui, X. Li, H. Ling, S. Vemuri, B. Olson, E. Li, Y. Zhang, J. Caverlee, D. Kalathil, and S. Ji. Curriculum reinforcement learning from easy to hard tasks improves llm reasoning. *arXiv preprint arXiv:2506.06632*, 2025.

Qwen Team, A. Yang, B. Yang, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023a.

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

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

N. Stiennon, L. Ouyang, J. Wu, D. M. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. Christiano. Learning to summarize from human feedback. In *NeurIPS*, 2020.

X. Tang, Y. Zhan, Z. Li, W. X. Zhao, Z. Zhang, Z. Wen, Z. Zhang, and J. Zhou. Rethinking sample polarity in reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2512.21625*, 2025.

Volcano Engine. VERL: Volcano Engine Reinforcement Learning for LLMs. `https://github.com/volcengine/verl`, 2024. Open-source implementation of the HybridFlow paper.

J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

X. Wen, Z. Liu, S. Zheng, S. Ye, Z. Wu, Y. Wang, Z. Xu, X. Liang, J. Li, Z. Miao, J. Bian, and M. Yang. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *arXiv preprint arXiv:2506.14245*, 2025.

E. Wiewiora, G. W. Cottrell, and C. Elkan. Principled methods for advising reinforcement learning agents. In *ICML*, pp. 792–799, 2003.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992.

A. Yang, A. Li, B. Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Q. Yu, Z. Zhang, R. Zhu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Y. Yue, Z. Chen, R. Lu, A. Zhao, Z. Yang, and H. Hu. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? In *NeurIPS*, 2025. Oral.

Y. Zeng, Z. Sun, B. Ji, E. Min, H. Cai, S. Wang, D. Yin, H. Zhang, X. Chen, and J. Wang. Cures: From gradient analysis to efficient curriculum learning for reasoning llms. *arXiv preprint arXiv:2510.01037*, 2025.

S. Zhang, G. Sun, K. Zhang, X. Guo, and R. Guo. Clpo: Curriculum learning meets policy optimization for llm reasoning. *arXiv preprint arXiv:2509.25004*, 2025.

X. Zhu, M. Xia, Z. Wei, W.-L. Chen, D. Chen, and Y. Meng. The surprising effectiveness of negative reinforcement in llm reasoning. In *NeurIPS*, 2025.

A Proof of Theorem 1 (Selective Regularization Decomposition)

Proof. This proof is for the unclipped on-policy actor term with detached group advantages, i.e., the setting described in Theorem 1 where $\rho_i = 1$ (equivalently, $\pi_{\text{old}} = \pi_\theta$, or clipping is inactive locally) and $\hat{\mu}_x$, $\hat{\sigma}_x$, and $\hat{A}^-(x)$ are treated as fixed with respect to θ. The gradient of $\widetilde{\mathcal{L}}_{\text{ACE}}$ differs from that of $\widetilde{\mathcal{L}}_{\text{std}}$ only in the negative advantage terms. We analyze the α-dependent component. Since $A^-_{\text{ACE},i} = \hat{A}^-_i \cdot (1 + \alpha \cdot \text{Softplus}(c_i))$, the additional gradient relative to the unclipped standard actor term is:

$$\Delta\nabla_\theta = -\alpha \mathbb{E}_{x\sim\mathcal{D}} \left[\frac{|\hat{A}^-(x)|}{G} \sum_{y_i \in \mathcal{Y}^-(x)} \text{Softplus}(c_i) \cdot \nabla_\theta \log \pi_\theta(y_i|x) \right] \tag{17}$$

Here $|\hat{A}^-(x)|$ is a per-prompt scalar (constant across rollouts within a group) that does not depend on y_i. This is the standard REINFORCE form: $|\hat{A}^-(x)| \cdot \text{Softplus}(c_i)$ acts as a *scalar reward* multiplying the score function, with c_i treated as not depending on θ (the "stop-gradient" convention standard in policy gradient methods).

As $G \to \infty$, by the law of large numbers:

$$\begin{aligned} \Delta\nabla_\theta &\to -\alpha \mathbb{E}_{x\sim\mathcal{D}} \left[|\hat{A}^-(x)| \sum_{y \in \mathcal{Y}^-(x)} \pi_\theta(y|x) \cdot \text{Softplus}(c(y)) \cdot \nabla_\theta \log \pi_\theta(y|x) \right] \\ &= -\alpha \mathbb{E}_{x\sim\mathcal{D}} \left[|\hat{A}^-(x)| \sum_{y \in \mathcal{Y}^-(x)} \text{Softplus}(c(y)) \cdot \nabla_\theta \pi_\theta(y|x) \right] \end{aligned} \tag{18}$$

using $\pi_\theta(y|x)\nabla_\theta \log \pi_\theta(y|x) = \nabla_\theta \pi_\theta(y|x)$.

Now, the true gradient of $\mathcal{R}_{\text{sel}}(\theta)$ requires differentiating $|\hat{A}^-(x)| \cdot \pi_\theta(y|x) \cdot \text{Softplus}(c(y))$ where $\pi_\theta(y|x)$ and $\text{Softplus}(c(y))$ both depend on θ (since $c(y) = \log \pi_\theta(y|x) - \log \pi_{\text{ref}}(y|x)$). Since $|\hat{A}^-(x)|$ is a per-prompt scalar, it factors out, and by the product rule:

$$\nabla_\theta \left[\pi_\theta(y|x) \cdot \text{Softplus}(c) \right] = \underbrace{\text{Softplus}(c) \cdot \nabla_\theta \pi_\theta(y|x)}_{\text{Term I: captured by ACE}} + \underbrace{\pi_\theta(y|x) \cdot \sigma(c) \cdot \nabla_\theta \log \pi_\theta(y|x)}_{\text{Term II: residual}} \tag{19}$$

where $\sigma(c) = \text{Softplus}'(c) = 1/(1 + e^{-c})$ and $\nabla_\theta c = \nabla_\theta \log \pi_\theta(y|x)$.

Multiplying by $|\hat{A}^-(x)|$, summing over $y \in \mathcal{Y}^-(x)$, and taking expectations, Eq. (18) matches exactly $-\alpha \cdot |\hat{A}^-(x)| \cdot$ Term I. Rearranging:

$$\Delta\nabla_\theta = -\alpha \nabla_\theta \mathcal{R}_{\text{sel}} + \alpha \underbrace{\mathbb{E}_{x\sim\mathcal{D}} \left[|\hat{A}^-(x)| \sum_{y \in \mathcal{Y}^-(x)} \pi_\theta(y|x)\, \sigma(c)\, \nabla_\theta \log \pi_\theta(y|x) \right]}_{\mathcal{E}(\theta)} \tag{20}$$

This is an *exact* identity with no approximation. The residual $\mathcal{E}(\theta)$ arises because ACE treats $\text{Softplus}(c_i)$ as a fixed reward signal, omitting the gradient through c_i itself. □

Remark 2 (Residual term and contrast with global KL)**.** *The residual $\mathcal{E}(\theta)$ is not negligible: for $c \in [1, 3]$, the ratio $\sigma(c)/\text{Softplus}(c)$ ranges from 31–56%. $\mathcal{E}$ arises because ACE treats* Softplus(c_i) *as a fixed scalar (stop-gradient), omitting the gradient that the full regularizer $\mathcal{R}_{\text{sel}}$ would contribute by differentiating* through *c_i (Term II in Eq. 19). This omitted gradient would suppress overconfident errors* more aggressively*: it drives the parameters to reduce not only $\pi_\theta(y|x)$ but also the confidence gap $c(y)$ itself. ACE therefore implements a* tempered *version of the full regularizer—correcting overconfident errors via the dominant Term I (confidence-weighted probability suppression) while forgoing Term II's sharper through-c correction.*

Remark 3 (Why stop-gradient is preferable to the full regularizer)**.** *A natural question is whether one should retain Term II to implement the full $\nabla_\theta \mathcal{R}_{\text{sel}}$ instead of ACE's tempered version. We argue against this for three reasons.* ***(i) Precedent for detaching θ-dependent signals.*** *Although the reward in vanilla REINFORCE does not depend on θ, modern policy gradient methods routinely stop-gradient through θ-dependent quantities used in the loss: PPO/GRPO detach the advantage $\hat{A}_i$ (computed from the current policy's rollouts) from the actor gradient; actor-critic methods detach the value baseline $V(s;\theta)$ even under parameter sharing; and the "old policy" π_{old} in importance ratios is always frozen. ACE's treatment of* $\text{Softplus}(c_i)$ *as a detached reward modifier follows the same principle: quantities that* diagnose *the policy state should inform gradient* magnitude*, not become optimization targets themselves.* ***(ii) Feedback loop.*** *Retaining Term II means the penalty magnitude itself becomes an optimization target: the gradient would simultaneously try to reduce $\pi_\theta(y|x)$* and *reduce $c_i = \log(\pi_\theta/\pi_{\text{ref}})$, creating a second-order feedback that can cause gradient oscillation and training instability.* ***(iii) Variance.*** *The gradient quality analysis (Theorem 2) proves that ACE's stop-gradient version improves the quality ratio Q_d under realistic conditions. Adding Term II introduces an additional score-function estimator $\sigma(c_i)\nabla_\theta \log \pi_\theta$, which increases gradient variance without a guaranteed commensurate signal gain—the sufficient condition for quality improvement (Eq. 49) would need to be re-derived and may no longer hold.*

B Gradient Quality Analysis

This appendix provides the full formal analysis of ACE's effect on gradient quality, summarized in §4.5.

Assumption 1. *For a fixed prompt x with pass rate p, let $g_i = A_i \nabla_\theta \log \pi_\theta(y_i|x)$ be the per-rollout gradient for incorrect rollouts ($r_i = 0$), and let $s_i = \nabla_\theta \log \pi_\theta(y_i|x)$ denote the score function. Let $\phi_i = \text{Softplus}(c_i)$. We assume:*

1. *Rollouts y_i are conditionally independent given x.*
2. *The signal direction is $\hat{d} = \mathbb{E}[s_i \mid r_i = 0]/\|\mathbb{E}[s_i \mid r_i = 0]\|$.*
3. *The directional covariance satisfies $\text{Cov}(\phi_i, (\hat{d}^\top s_i)^2 \mid r_i = 0) > 0$, i.e., overconfident errors tend to have score functions more aligned with the expected gradient direction.*

Proposition 1 (Second Moment Increase)**.** *For any $\alpha > 0$, ACE strictly increases the mean squared gradient norm of incorrect rollouts:*

$$\mathbb{E}[\|g_i^{\text{ACE}}\|^2 \mid r_i = 0] > \mathbb{E}[\|g_i^{\text{std}}\|^2 \mid r_i = 0] \tag{21}$$

whenever $\mathbb{E}[\phi_i \|s_i\|^2 \mid r_i = 0] > 0$ (i.e., errors are not all zero-gradient). This is an unavoidable consequence of the purely additive penalty structure: $(1 + \alpha\phi_i) > 1$ for all $\phi_i > 0$.

Proof. Let $a = |\hat{A}^-(x)| > 0$ denote the per-prompt base penalty magnitude. Under standard GRPO: $g_i^{\text{std}} = a \cdot s_i$. Under ACE: $g_i^{\text{ACE}} = a(1 + \alpha\phi_i) \cdot s_i$. Then:

$$\begin{aligned}
\mathbb{E}[\|g_i^{\text{ACE}}\|^2] - \mathbb{E}[\|g_i^{\text{std}}\|^2] &= a^2 \left(\mathbb{E}[(1 + \alpha\phi_i)^2 \|s_i\|^2] - \mathbb{E}[\|s_i\|^2]\right) \\
&= a^2 \left(\underbrace{2\alpha\, \mathbb{E}[\phi_i \|s_i\|^2]}_{>0} + \underbrace{\alpha^2\, \mathbb{E}[\phi_i^2 \|s_i\|^2]}_{\geq 0} \right) > 0
\end{aligned} \tag{22}$$

since $a > 0$, $\phi_i = \text{Softplus}(c_i) > 0$, $\alpha > 0$, and $\|s_i\|^2 \geq 0$ with $\mathbb{E}[\phi_i \|s_i\|^2] > 0$. □

Definition 3 (Directional Signal and Variance)**.** *For incorrect rollouts, let $\hat{d} = \mathbb{E}[s_i \mid r_i = 0]/\|\mathbb{E}[s_i \mid r_i = 0]\|$ be the unit vector along the expected score function. The* ***directional signal*** *and* ***directional variance*** *of a gradient estimator $g_i = w_i \cdot s_i$ are:*

$$\mu_d = \mathbb{E}[\hat{d}^\top g_i \mid r_i = 0] \qquad \textit{(signal along } \hat{d}\textit{)} \tag{23}$$

$$\sigma_d^2 = \text{Var}[\hat{d}^\top g_i \mid r_i = 0] \qquad \textit{(noise along } \hat{d}\textit{)} \tag{24}$$

The ***gradient quality ratio*** *is $Q_d = \mu_d^2/\sigma_d^2$.*

Theorem 2 (Improved Gradient Quality via ACE)**.** *Under Assumption 1, let $\hat{d}$ be the signal direction. Define the directional projections $u_i = \hat{d}^\top s_i$ (scalar random variables). Assume:*

$$\mathrm{Cov}(\phi_i,\, u_i^2 \mid r_i = 0) > 0 \tag{25}$$

i.e., overconfident errors tend to have score functions more aligned with the expected gradient direction. Then:

(a) Directional variance increase. *For any $\alpha > 0$, ACE increases the directional variance:*

$$\mathrm{Var}[\hat{d}^\top g_i^{\mathrm{ACE}} \mid r_i = 0] > \mathrm{Var}[\hat{d}^\top g_i^{\mathrm{std}} \mid r_i = 0] \tag{26}$$

whenever $\mathrm{Cov}(\phi_i, u_i^2) > 0$ and $\mathbb{E}[\phi_i] > 0$. This is an unavoidable consequence of the additive reweighting structure, analogous to the total second-moment increase (Proposition 1).

(b) Quality improvement under high-variance conditions. *Assume additionally that the initial gradient is noisy relative to the signal, i.e., $\mathrm{Var}[u_i] > (\mathbb{E}[u_i])^2$ (equivalently, $Q_d^{\mathrm{std}} < 1$). Then for sufficiently small $\alpha > 0$, the gradient quality ratio of ACE strictly dominates that of standard GRPO:*

$$Q_d^{\mathrm{ACE}} > Q_d^{\mathrm{std}} \tag{27}$$

Consequently, although ACE increases both the signal and the noise (directional variance), the signal grows faster, yielding a net improvement in gradient quality along the optimization-relevant direction.

Proof. Consider a fixed prompt x with G rollouts sampled i.i.d. from $\pi_\theta(\cdot|x)$. Let $s_i = \nabla_\theta \log \pi_\theta(y_i|x)$ denote the score function for rollout y_i, and let $\phi_i = \mathrm{Softplus}(c_i)$. All expectations below are conditioned on $r_i = 0$.

Setup and notation. Let $a = |\hat{A}^-(x)| > 0$ denote the per-prompt base penalty magnitude. Under standard GRPO: $g_i^{\mathrm{std}} = a \cdot s_i$. Under ACE: $g_i^{\mathrm{ACE}} = a(1 + \alpha\phi_i)s_i$. Since a is a positive scalar constant (per-prompt), it cancels in the gradient quality ratio $Q_d = \mu_d^2/\sigma_d^2$. We therefore analyze the normalized weights $w_i^{\mathrm{std}} = 1$ and $w_i^{\mathrm{ACE}} = 1 + \alpha\phi_i$ without loss of generality. Let $\hat{d} = \mathbb{E}[s_i]/\|\mathbb{E}[s_i]\|$ be the signal direction, and define the scalar projections $u_i = \hat{d}^\top s_i$.

Step 1: Directional variance analysis (Part (a)). The directional variance is:

$$\mathrm{Var}[w_i u_i] = \mathbb{E}[w_i^2 u_i^2] - (\mathbb{E}[w_i u_i])^2 \tag{28}$$

For standard GRPO ($w_i = 1$): $\mathrm{Var}^{\mathrm{std}} = \mathbb{E}[u_i^2] - (\mathbb{E}[u_i])^2$.

For ACE ($w_i = 1 + \alpha\phi_i$):

$$\mathbb{E}[w_i^2 u_i^2] = \mathbb{E}[u_i^2] + 2\alpha\,\mathbb{E}[\phi_i u_i^2] + \alpha^2\,\mathbb{E}[\phi_i^2 u_i^2] \tag{29}$$

$$\begin{aligned}(\mathbb{E}[w_i u_i])^2 &= (\mathbb{E}[u_i] + \alpha\,\mathbb{E}[\phi_i u_i])^2 \\ &= (\mathbb{E}[u_i])^2 + 2\alpha\,\mathbb{E}[u_i]\,\mathbb{E}[\phi_i u_i] + \alpha^2(\mathbb{E}[\phi_i u_i])^2\end{aligned} \tag{30}$$

Subtracting Eq. (30) from Eq. (29):

$$\mathrm{Var}^{\mathrm{ACE}} = \mathrm{Var}^{\mathrm{std}} + 2\alpha \underbrace{\left(\mathbb{E}[\phi_i u_i^2] - \mathbb{E}[u_i]\,\mathbb{E}[\phi_i u_i]\right)}_{\equiv \Delta_1} + O(\alpha^2) \tag{31}$$

Step 2: Sign of Δ_1. Decompose using identities:

$$\mathbb{E}[\phi_i u_i^2] = \mathrm{Cov}(\phi_i, u_i^2) + \mathbb{E}[\phi_i]\,\mathbb{E}[u_i^2] \tag{32}$$

$$\mathbb{E}[\phi_i u_i] = \mathrm{Cov}(\phi_i, u_i) + \mathbb{E}[\phi_i]\,\mathbb{E}[u_i] \tag{33}$$

Substituting into Δ_1:

$$\begin{aligned}\Delta_1 &= \mathrm{Cov}(\phi_i, u_i^2) + \mathbb{E}[\phi_i]\,\mathbb{E}[u_i^2] - \mathbb{E}[u_i]\,(\mathrm{Cov}(\phi_i, u_i) + \mathbb{E}[\phi_i]\,\mathbb{E}[u_i]) \\ &= \mathrm{Cov}(\phi_i, u_i^2) - \mathbb{E}[u_i]\,\mathrm{Cov}(\phi_i, u_i) + \mathbb{E}[\phi_i]\,\mathrm{Var}[u_i] \end{aligned} \tag{34}$$

The third term $\mathbb{E}[\phi_i]\,\mathrm{Var}[u_i] > 0$ always increases variance. In fact, $\Delta_1 > 0$ under typical conditions: for the natural Gaussian linear model where $u_i \sim \mathcal{N}(\mu, \sigma^2)$ and $\phi_i = a + bu_i$ ($a > 0$, $b > 0$), we have $\mathrm{Cov}(\phi_i, u_i^2) = 2b\mu\sigma^2$, $\mathrm{Cov}(\phi_i, u_i) = b\sigma^2$, and $\mathbb{E}[\phi_i] = a + b\mu$, giving:

$$\Delta_1 = 2b\mu\sigma^2 - \mu \cdot b\sigma^2 + (a + b\mu)\sigma^2 = 2b\mu\sigma^2 + a\sigma^2 > 0 \tag{35}$$

This confirms that **directional variance increases** under ACE—an unavoidable cost of additive reweighting. The condition $\Delta_1 < 0$ would require:

$$\mathbb{E}[u_i]\,\mathrm{Cov}(\phi_i, u_i) > \mathrm{Cov}(\phi_i, u_i^2) + \mathbb{E}[\phi_i]\,\mathrm{Var}[u_i] \tag{36}$$

which is violated in the Gaussian linear model and is difficult to satisfy in practice. However, as we show next, this does *not* prevent quality improvement: what matters is that the signal grows faster than the square root of variance.

Step 3: Quality improvement (Part (b)). The gradient quality ratio is:

$$Q_d = \frac{(\mathbb{E}[w_i u_i])^2}{\mathrm{Var}[w_i u_i]} \tag{37}$$

Since $\Delta_1 > 0$ in general (Step 2), the directional variance increases. Nevertheless, the quality ratio can still improve because ACE also increases the signal $\mathbb{E}[w_i u_i]$. We now provide the complete derivation.

Signal computation. For ACE with $w_i = 1 + \alpha\phi_i$:

$$\begin{aligned}\mu^{\mathrm{ACE}} \triangleq \mathbb{E}[w_i u_i] &= \mathbb{E}[(1 + \alpha\phi_i)u_i] = \mathbb{E}[u_i] + \alpha\,\mathbb{E}[\phi_i u_i] \\ &= \mathbb{E}[u_i] + \alpha\,(\mathrm{Cov}(\phi_i, u_i) + \mathbb{E}[\phi_i]\,\mathbb{E}[u_i]) \\ &= \mathbb{E}[u_i](1 + \alpha\,\mathbb{E}[\phi_i]) + \alpha\,\mathrm{Cov}(\phi_i, u_i) \end{aligned} \tag{38}$$

Denote $\mu \triangleq \mathbb{E}[u_i]$, $\bar{\phi} \triangleq \mathbb{E}[\phi_i]$, and $C \triangleq \mathrm{Cov}(\phi_i, u_i)$. Then:

$$\mu^{\mathrm{ACE}} = \mu(1 + \alpha\bar{\phi}) + \alpha C \tag{39}$$

The squared signal is:

$$\begin{aligned}(\mu^{\mathrm{ACE}})^2 &= \big(\mu(1 + \alpha\bar{\phi}) + \alpha C\big)^2 \\ &= \mu^2(1 + \alpha\bar{\phi})^2 + 2\alpha C\mu(1 + \alpha\bar{\phi}) + \alpha^2 C^2 \\ &= \mu^2 + 2\alpha\mu^2\bar{\phi} + \alpha^2\mu^2\bar{\phi}^2 + 2\alpha C\mu + 2\alpha^2 C\mu\bar{\phi} + \alpha^2 C^2 \\ &= \mu^2 + 2\alpha(\mu^2\bar{\phi} + C\mu) + O(\alpha^2) \end{aligned} \tag{40}$$

Variance computation. From Step 2, we have:

$$\mathrm{Var}^{\mathrm{ACE}} = \mathrm{Var}^{\mathrm{std}} + 2\alpha\,\Delta_1 + O(\alpha^2) \tag{41}$$

where $\mathrm{Var}^{\mathrm{std}} = \mathbb{E}[u_i^2] - \mu^2$ and Δ_1 is given by Eq. (34).

Quality ratio expansion. We compute the difference in quality ratios. For standard GRPO:

$$Q_d^{\mathrm{std}} = \frac{\mu^2}{\mathrm{Var}^{\mathrm{std}}} \tag{42}$$

For ACE, using Eqs. (40) and (41):

$$Q_d^{\mathrm{ACE}} = \frac{(\mu^{\mathrm{ACE}})^2}{\mathrm{Var}^{\mathrm{ACE}}} = \frac{\mu^2 + 2\alpha(\mu^2\bar{\phi} + C\mu) + O(\alpha^2)}{\mathrm{Var}^{\mathrm{std}} + 2\alpha\,\Delta_1 + O(\alpha^2)} \tag{43}$$

Using the first-order Taylor expansion $(1+x)^{-1} \approx 1 - x$ for small x:

$$\begin{aligned} Q_d^{\mathrm{ACE}} &= \frac{\mu^2 + 2\alpha(\mu^2\bar{\phi} + C\mu)}{\mathrm{Var}^{\mathrm{std}}} \left(1 - \frac{2\alpha\,\Delta_1}{\mathrm{Var}^{\mathrm{std}}}\right) + O(\alpha^2) \\ &= \frac{\mu^2}{\mathrm{Var}^{\mathrm{std}}} + \frac{2\alpha(\mu^2\bar{\phi} + C\mu)}{\mathrm{Var}^{\mathrm{std}}} - \frac{2\alpha\mu^2\,\Delta_1}{(\mathrm{Var}^{\mathrm{std}})^2} + O(\alpha^2) \\ &= Q_d^{\mathrm{std}} + \frac{2\alpha}{\mathrm{Var}^{\mathrm{std}}} \left(\mu^2\bar{\phi} + C\mu - \frac{\mu^2}{\mathrm{Var}^{\mathrm{std}}}\,\Delta_1\right) + O(\alpha^2) \end{aligned} \tag{44}$$

Therefore:

$$Q_d^{\mathrm{ACE}} - Q_d^{\mathrm{std}} = \frac{2\alpha}{\mathrm{Var}^{\mathrm{std}}} \underbrace{\left(\mu^2\bar{\phi} + C\mu - Q_d^{\mathrm{std}}\,\Delta_1\right)}_{\triangleq\,\Gamma} + O(\alpha^2) \tag{45}$$

Sufficient condition for improvement. Quality improves when $\Gamma > 0$. Substituting Δ_1 from Eq. (34):

$$\begin{aligned} \Gamma &= \mu^2\bar{\phi} + C\mu - Q_d^{\mathrm{std}} \left(\mathrm{Cov}(\phi_i, u_i^2) - \mu\,C + \bar{\phi}\,\mathrm{Var}^{\mathrm{std}}\right) \\ &= \mu^2\bar{\phi} + C\mu - Q_d^{\mathrm{std}}\,\mathrm{Cov}(\phi_i, u_i^2) + Q_d^{\mathrm{std}}\mu\,C - Q_d^{\mathrm{std}}\bar{\phi}\,\mathrm{Var}^{\mathrm{std}} \end{aligned} \tag{46}$$

Using $Q_d^{\mathrm{std}} = \mu^2/\mathrm{Var}^{\mathrm{std}}$, the last term becomes $-\mu^2\bar{\phi}$, which cancels with the first term:

$$\begin{aligned} \Gamma &= C\mu + Q_d^{\mathrm{std}}\mu\,C - Q_d^{\mathrm{std}}\,\mathrm{Cov}(\phi_i, u_i^2) \\ &= C\mu(1 + Q_d^{\mathrm{std}}) - Q_d^{\mathrm{std}}\,\mathrm{Cov}(\phi_i, u_i^2) \end{aligned} \tag{47}$$

Under Assumption 1, $C = \mathrm{Cov}(\phi_i, u_i) > 0$ and $\mathrm{Cov}(\phi_i, u_i^2) > 0$ (overconfident errors have gradients more aligned with the signal direction). We analyze $\Gamma > 0$:

$$\Gamma > 0 \quad\Longleftrightarrow\quad C\mu(1 + Q_d^{\mathrm{std}}) > Q_d^{\mathrm{std}}\,\mathrm{Cov}(\phi_i, u_i^2) \tag{48}$$

Rearranging:

$$\frac{C\mu}{\mathrm{Cov}(\phi_i, u_i^2)} > \frac{Q_d^{\mathrm{std}}}{1 + Q_d^{\mathrm{std}}} \tag{49}$$

The right-hand side is a monotonically increasing function of Q_d^{std} that ranges from 0 (when $Q_d^{\mathrm{std}} = 0$) to 1 (as $Q_d^{\mathrm{std}} \to \infty$). Therefore, when $Q_d^{\mathrm{std}} < 1$ (equivalently, $\mathrm{Var}[u_i] > (\mathbb{E}[u_i])^2$), we have:

$$\frac{Q_d^{\mathrm{std}}}{1 + Q_d^{\mathrm{std}}} < \frac{1}{2} \tag{50}$$

Under the Gaussian linear model ($u_i \sim \mathcal{N}(\mu, \sigma^2)$, $\phi_i = a + bu_i$), we can verify:

$$C = \mathrm{Cov}(\phi_i, u_i) = b\sigma^2 \tag{51}$$

$$\mathrm{Cov}(\phi_i, u_i^2) = b\,\mathrm{Cov}(u_i, u_i^2) = b \cdot 2\mu\sigma^2 = 2b\mu\sigma^2 \tag{52}$$

Thus:

$$\frac{C\mu}{\mathrm{Cov}(\phi_i, u_i^2)} = \frac{b\sigma^2 \cdot \mu}{2b\mu\sigma^2} = \frac{1}{2} \tag{53}$$

Combined with Eq. (49), when $Q_d^{\mathrm{std}} < 1$:

$$\frac{1}{2} > \frac{Q_d^{\mathrm{std}}}{1 + Q_d^{\mathrm{std}}} \quad\Longrightarrow\quad \Gamma > 0 \quad\Longrightarrow\quad Q_d^{\mathrm{ACE}} > Q_d^{\mathrm{std}} \tag{54}$$

This completes the proof that quality improves under the high-variance condition:

$$\mathrm{Var}[u_i] > (\mathbb{E}[u_i])^2 \qquad \Longleftrightarrow \qquad Q_d^{\mathrm{std}} < 1 \tag{55}$$

This high-variance regime is the typical operating condition in stochastic policy gradient optimization, where individual rollout gradients are highly variable. Under this condition, the signal growth term dominates the variance growth term, ensuring $Q_d^{\mathrm{ACE}} > Q_d^{\mathrm{std}}$.

Summary. ACE's confidence-dependent weighting increases both the total gradient second moment (Proposition 1) and the directional variance (Steps 1–2)—both unavoidable consequences of additive reweighting. However, ACE improves the gradient quality ratio (Step 3) under two conditions: (i) overconfident errors carry gradient signal aligned with the optimization direction ($\mathrm{Cov}(\phi_i, u_i) > 0$), and (ii) the initial quality ratio is low ($\mathrm{Var}[u_i] > (\mathbb{E}[u_i])^2$). The key mechanism is that ACE's selective amplification of high-confidence errors concentrates extra weight on the most informative gradients, causing the signal to grow faster than the noise along the optimization-relevant direction. □

C Implementation Details

Sequence-level vs. token-level aggregation. While Definition 1 defines c_i at the sequence level, one can also define a token-level variant $c_i^{(t)}$ and apply ACE per-token. We use the sequence-level aggregation $c_i = \sum_t c_i^{(t)}$ in our main experiments to capture "trajectory confidence."

Compute overhead. ACE adds exactly one Softplus computation per incorrect rollout per training step. Given that the bottleneck of RLVR training is rollout generation (model inference), the overhead of ACE is negligible ($< 0.1\%$ of wall-clock time).

PyTorch implementation sketch.

```
def ace_advantage(rewards, log_probs_policy, log_probs_ref, alpha=1.0):
    """
    Args:
        rewards: (B, G) binary rewards
        log_probs_policy: (B, G) sequence-level log probs under pi_theta
        log_probs_ref: (B, G) sequence-level log probs under pi_ref
        alpha: ACE strength
    Returns:
        advantages: (B, G) modified advantages
    """
    # Standard group statistics
    pass_rate = rewards.mean(dim=-1, keepdim=True)  # (B, 1)
    std = rewards.std(dim=-1, keepdim=True) + 1e-8
    std_advantage = (rewards - pass_rate) / std  # (B, G)

    # Confidence score (already available, zero extra compute)
    c = log_probs_policy - log_probs_ref  # (B, G)
    # Normalize by sequence length
    c = c / seq_lengths

    # ACE advantage for negative samples
    ace_neg = std_advantage * (1.0 + alpha * F.softplus(c))  # (B, G)

    # Combine: use standard advantage for correct, ACE for incorrect
    is_correct = (rewards == 1).float()
    advantages = is_correct * std_advantage + (1 - is_correct) * ace_neg
```

```
    return advantages
```

Compatibility. The above can be dropped into any RLVR training loop that uses GRPO, PPO, or REINFORCE by replacing the advantage computation. No changes are needed to the model architecture, rollout generation, or reward computation.

D Sensitivity to α

We vary $\alpha \in \{0, 0.1, 0.5, 1.0, 2.0, 5.0\}$ on MATH-500 using Qwen2.5-Math-7B ($\alpha = 0$ recovers standard GRPO). We conduct the sensitivity analysis on a single model to isolate the effect of α; since the main results (Tables 1–2) demonstrate consistent gains across all three model families at $\alpha = 1.0$, we expect the optimal range to transfer.

Table 4: Sensitivity to α on MATH-500 (Qwen2.5-Math-7B). α=1.0 achieves optimal Pass@32 while preserving Pass@1.

α	Pass@1 (%)	Pass@32 (%)
0.0 (GRPO)	73.4	91.3
0.1	73.5	91.9
0.5	73.8	93.2
1.0 (default)	**74.2**	**94.3**
2.0	73.5	93.5
5.0	72.4	92.0

Observations. As shown in Table 4, optimal performance is achieved at $\alpha = 1.0$ with Pass@32 = 94.3% (+3.0pp over GRPO). Performance remains stable across $\alpha \in [0.5, 2.0]$, all outperforming standard GRPO. Pass@1 shows a slight decrease only at larger α values (≥ 2.0), reflecting the exploration-exploitation trade-off. We adopt $\alpha = 1.0$ as the default for all experiments.

E Training Hyperparameters

Table 5 summarizes the training hyperparameters for all three models.

Fair comparison (matched recipe within each model). To ensure improvements are attributable to ACE rather than tuning differences, we use the *same* training recipe and training budget for all methods *within a given model* (GRPO vs. ACE-GRPO, and DAPO vs. ACE-DAPO). Concretely, for a fixed model we keep the data, verifier, rollout group size G, sampling settings, optimizer, learning rate schedule, batch sizes, clipping/KL coefficients, maximum sequence lengths, and the number of optimizer updates identical across methods; ACE changes only the computation of the negative advantages through Eq. (5) (controlled by α). Hyperparameters may differ *across* model families due to model-specific stability and context-length constraints, but cross-method comparisons are always performed under matched settings for the same model.

Table 5: Training hyperparameters for ACE-GRPO experiments.

Hyperparameter	**Qwen2.5-Math-7B**	**Qwen3-8B-Base**	**Llama-3.1-8B-Instruct**
Total epochs	10	10	10
Training batch size	2048	1024	1024
Mini-batch size	1024	1024	1024
Micro-batch size per GPU	16	16	16
Learning rate	1×10^{-5}	5×10^{-7}	1×10^{-6}
Optimizer	AdamW	AdamW	AdamW
Temperature	1.0	1.0	1.0
Max prompt length	1024	1024	1024
Max response length	3000	8192	4096
Rollout samples per prompt	8	8	8
Validation samples	128	128	128
GPU memory utilization	0.75	0.75	0.75
KL coefficient β	0.001	0.001	0.001
Enable thinking (Qwen3)	–	False	–

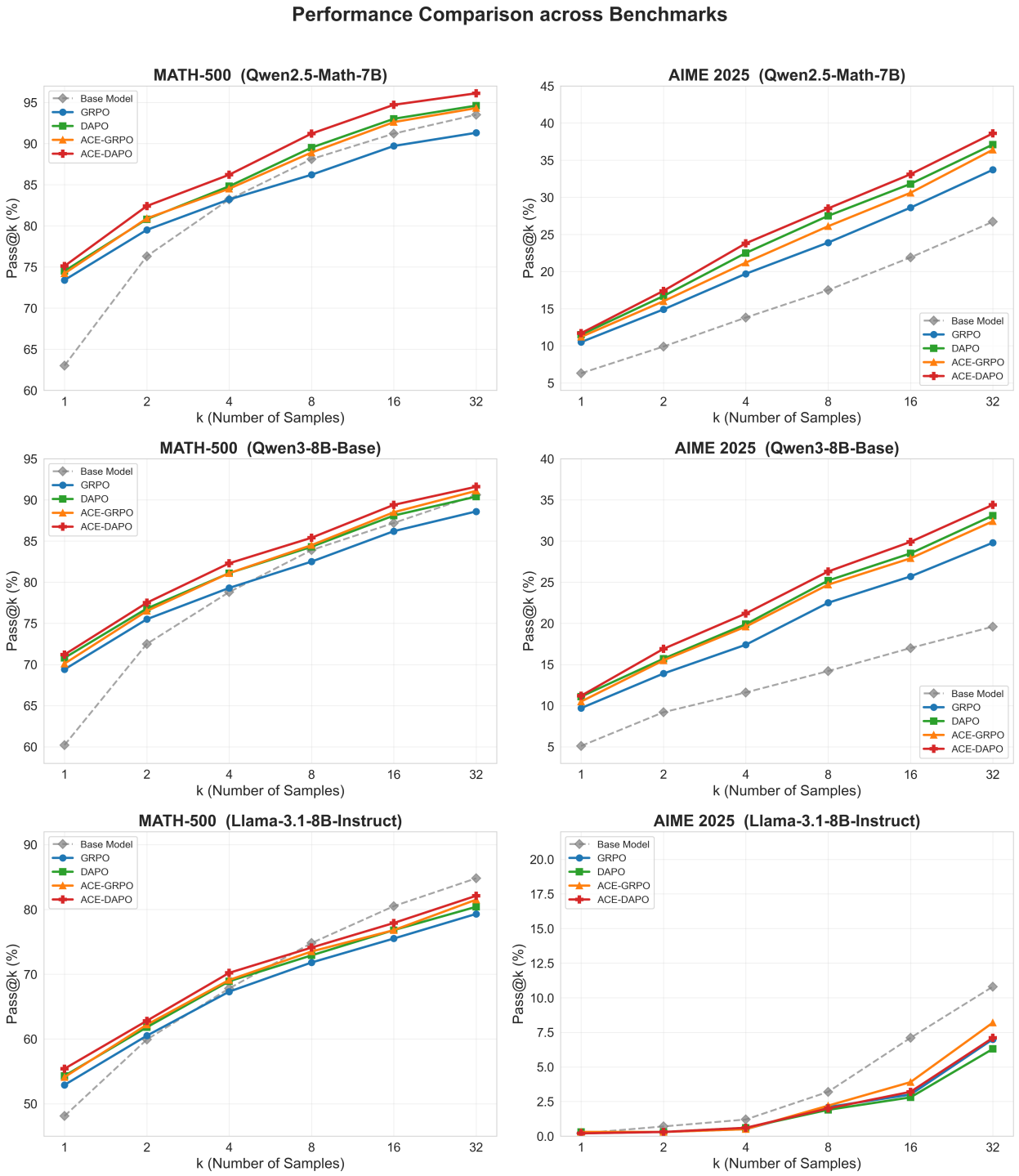

Figure 2: **Performance Comparison across Benchmarks.** Pass@k curves for all five methods on MATH-500 (left column) and AIME 2025 (right column) across three model families: Qwen2.5-Math-7B (top row), Qwen3-8B-Base (middle row), and Llama-3.1-8B-Instruct (bottom row). ACE-GRPO and ACE-DAPO consistently outperform their respective baselines on the two Qwen families, with larger gains at higher k values. On Llama-3.1-8B-Instruct, ACE-GRPO improves large-k performance while ACE-DAPO is mixed on AIME 2025, indicating weaker but still informative cross-family transfer.

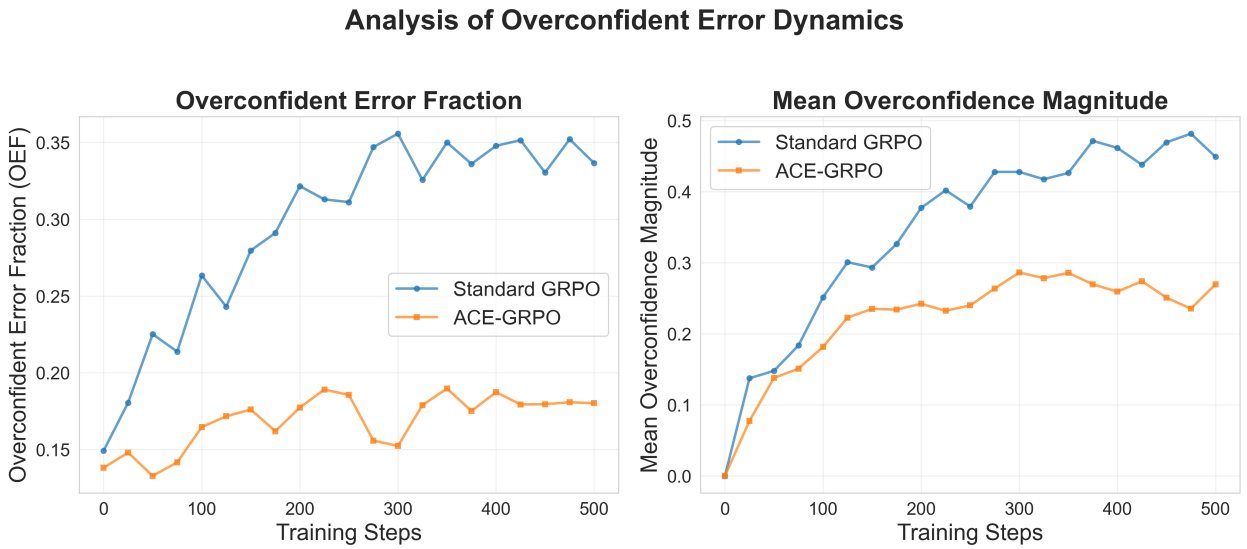

Figure 3: **Overconfident Error Dynamics.** Left: Overconfident error fraction (OEF) over training. Right: Mean overconfidence magnitude for $c_i > 0$ errors. ACE-GRPO effectively suppresses both metrics compared to standard GRPO.

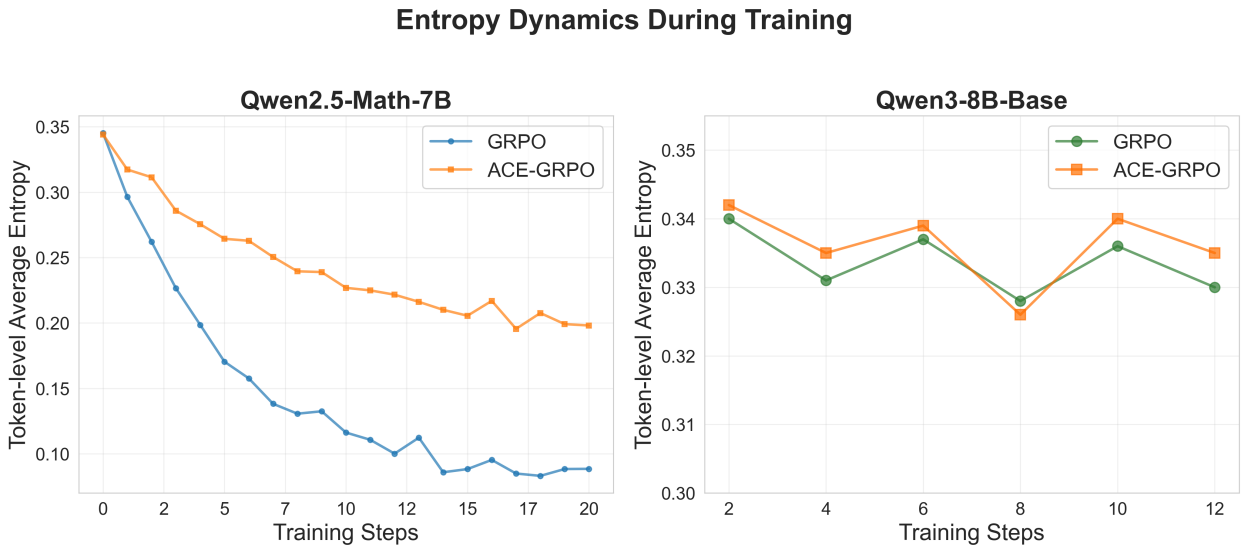

Figure 4: **Entropy Dynamics.** Token-level entropy over the first 20 training steps. Left: On Qwen2.5-Math-7B, ACE-GRPO retains substantially more entropy than standard GRPO, which suffers rapid entropy collapse. Right: On Qwen3-8B-Base, ACE-GRPO maintains more stable entropy, demonstrating consistency across architectures. We report entropy dynamics for the two Qwen models only; Llama-3.1-8B-Instruct is excluded because its lower baseline accuracy makes the entropy signal less directly comparable (see §5 for discussion).