

C²GSPG: CONFIDENCE-CALIBRATED GROUP SEQUENCE POLICY GRADIENT TOWARDS SELF-AWARE REASONING

Anonymous authors

Paper under double-blind review

ABSTRACT

Reinforcement Learning (RL) methods, exemplified by Group Relative Policy Optimization (GRPO) and its variants, play a central role in developing reasoning models. However, these methods often suffer from a critical overconfidence issue, which prevents them from achieving self-aware reasoning models. In this study, we propose a simple yet effective confidence-calibration group sequence policy gradient method, called C²GSPG, which simultaneously enhances reasoning performance while suppressing overconfidence. In principle, we propose a Group Sequence Policy Gradient (GSPG) framework for learning reasoning models, which eliminates the token-level bias commonly appearing in GRPO and its variants. In this framework, we define the model confidence for each reasoning problem using the normalized sequence-level probability, and then apply a cross-entropy regularizer to calibrate the model confidence to the sequence’s reward. We demonstrate that the confidence calibration regularizer and GSPG are collaborative for binary rewards, as their objectives always share the same gradient direction. For non-binary rewards, we apply nonlinear reward normalization and adaptive regularizer clipping, mitigating the potential conflict between the two objectives. Applying C²GSPG to post-train large language models in logical and mathematical reasoning tasks, we show its superiority over state-of-the-art methods in both reasoning accuracy and confidence calibration.

1 INTRODUCTION

Reinforcement Learning (RL) has become a pivotal post-training technique for enhancing the advanced reasoning capabilities of Large Language Models (LLMs). Unlike traditional Supervised Fine-Tuning (SFT), RL-based methods utilize verifiable rewards to enable models to explore and refine their reasoning pathways autonomously. Among prominent RL algorithms, the recent Group Relative Policy Optimization (GRPO) (Shao et al., 2024) has garnered significant attention. Unlike PPO (Schulman et al., 2017) and DPO (Rafailov et al., 2023), GRPO eliminates the need for a value model, thereby simplifying model design and improving computational efficiency. Along this direction, many variants of GRPO have been proposed, e.g., DAPO (Yu et al., 2025), GPG (Chu et al., 2025), and GSPO (Zheng et al., 2025), and demonstrated superior performance in logical and mathematical reasoning tasks.

While GRPO and its variants can improve the reasoning ability of LLMs, they often induce an *overconfidence* issue — the post-trained models cannot be aware of their mistakes (Bereket & Leskovec, 2025; Chhikara, 2025). In the paradigm of RL, this issue is reflected by the mismatch between the probabilities of generated sequences and the rewards associated with the sequences, i.e., the high-probability sequences may lead to low rewards, as shown in Figure 1(a). To mitigate the overconfidence issue, some methods first modulate reward functions based on empirical or learnable model confidences, and then reduce the significance of high-confidence but low-quality samples during training (Zhou et al., 2025; Damani et al., 2025; Wen et al., 2024; Zeng et al., 2025). However, these methods often suffer from degraded reasoning accuracy or insufficient confidence calibration because they ignore the potential conflicts between policy optimization and confidence calibration. How to pursue confidence calibration without compromising policy optimization, especially in scenarios with non-binary rewards, remains an open problem.

054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
100
101
102
103
104
105
106
107

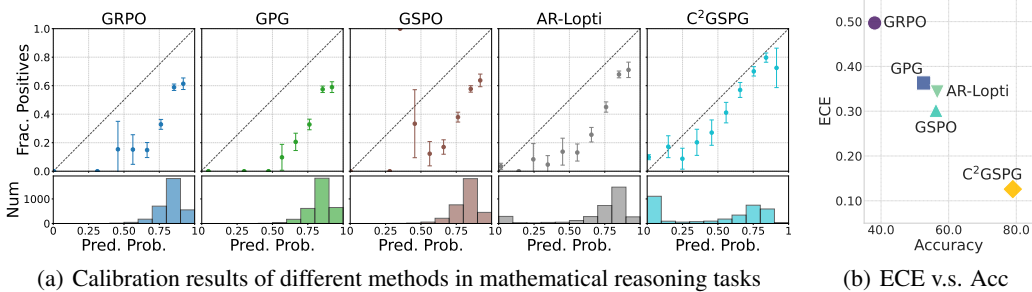


Figure 1: The comparison for various methods on the consistency between model confidences and rewards. The panel (a) presents reliability diagrams of various methods on six mathematical reasoning tasks, demonstrating our method’s effective calibration. The panel (b) shows Expected Calibration Error (ECE) against Accuracy on the “Knights and Knaves” logic puzzle (K&K) dataset (Xie et al., 2024), where our method reaches the best performance quadrant (high accuracy and low ECE).

In this study, we propose a simple yet effective RL method, called Confidence-calibrated Group Sequence Policy Gradient (C²GSPG), which yields robust reasoning models with calibrated confidence. For each reasoning problem, our method generates a group of sequences and defines model confidences of the sequences using normalized sequence-level probabilities. Based on the confidence, we propose a confidence-modulated advantage function and derive a Group Sequence Policy Gradient (GSPG) framework for training reasoning models. Similar to existing sequence-based RL methods (Zheng et al., 2025), GSPG eliminates the token-level bias of GRPO during training (Yang et al., 2025b). Furthermore, we introduce a Binary Cross-Entropy (BCE) regularizer into the framework to encourage the consistency between each sequence’s model confidence and the corresponding reward, resulting in the proposed C²GSPG method. We theoretically demonstrate a collaborative relationship between policy optimization and our regularizer in binary-reward scenarios — these two objectives share the same gradient direction in all situations. This insight motivates us to propose nonlinear reward normalization and adaptive regularizer clipping to mitigate the potential conflict between the two objectives in the more complex, non-binary reward cases.

We compare C²GSPG with GRPO and its variants on logical and mathematical reasoning tasks. The results demonstrate that our method effectively mitigates the model overconfidence problem while also improving reasoning ability. Moreover, Figure 1(a) validates that, unlike baselines which often exhibit poorly separated distributions, C²GSPG successfully cultivates a bimodal confidence profile: high confidence is predominantly assigned to correct answers, while incorrect answers are given low confidence. Furthermore, Figure 1(b) illustrates that C²GSPG consistently achieves a higher accuracy on the validation set while simultaneously driving down the Expected Calibration Error (ECE) more effectively than competing approaches.

2 PRELIMINARIES AND RELATED WORK

2.1 GROUP RELATIVE POLICY OPTIMIZATION AND ITS VARIANTS

RL-based fine-tuning has become a cornerstone for improving the reasoning ability of LLMs (Yang et al., 2025a; Shao et al., 2024; Comanici et al., 2025; Jaech et al., 2024). In the context of RL, an LLM is often represented as a parametric policy, denoted as π_θ . Given the current policy, denoted as $\pi_{\theta_{\text{old}}}$, **GRPO** (Shao et al., 2024) samples a group of G responses, denoted as $\{o_i\}_{i=1}^G$, for each prompt q according to the distribution $\pi_{\theta_{\text{old}}}(\cdot|q)$. Each response is a sequence of tokens (i.e., $o_i = \{o_{i,t}\}_{t=1}$, and $o_{i,t}$ is the t -th token of o_i), which includes a step-by-step reasoning chain leading to a final answer. A scalar reward r_i is assigned to each response o_i . Instead of applying the reward directly, GRPO computes a relative advantage for each response by comparing it to other responses sampled for the same prompt, i.e.,

$$\hat{A}_i = \frac{r_i - m}{\sigma}, \text{ where } m = \frac{1}{G} \sum_{i=1}^G r_i \text{ and } \sigma = \sqrt{\frac{1}{G} \sum_{i=1}^G (r_i - m)^2}. \quad (1)$$

GRPO incorporates this relative advantage into a clipped surrogate objective, which constrains the magnitude of policy updates. The overall objective function to be maximized is:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \min \left[\frac{\pi_{\theta, i, t}}{\pi_{\theta_{\text{old}}, i, t}} \hat{A}_i, \text{clip}_{\epsilon} \left(\frac{\pi_{\theta, i, t}}{\pi_{\theta_{\text{old}}, i, t}} \right) \hat{A}_i \right] - \gamma D_{\text{KL}}(\pi_{\theta} \| \pi_{\text{ref}}). \quad (2)$$

Here, we denote $\pi_{\theta}(o_{i,t} | o_{i, < t}, q)$ as $\pi_{\theta, i, t}$ for short. $|o_i|$ represents the length of o_i . The function $\text{clip}_{\epsilon}(\cdot) = \min(\max(\cdot, 1 - \epsilon), 1 + \epsilon)$, and π_{ref} denotes a reference policy (e.g., the pre-trained LLM).

Recently, some efforts have been made to improve GRPO (Chu et al., 2025; Zheng et al., 2025; Yu et al., 2025; Yang et al., 2025b; Li et al., 2025). Yang et al. (2025b) uses Advantage Reweighting and Low-Probability Token Isolation (**AR-Lopti**) to mitigate the over-dominance of low-probability tokens during training. Group Policy Gradient (**GPG**) (Chu et al., 2025) updates the policy by maximizing the logarithmic token probabilities weighted by unnormalized advantages, i.e.,

$$\mathcal{J}_{\text{GPG}}(\theta) = \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \frac{1}{\sum_{j=1}^G |o_j|} \sum_{i=1}^G \sum_{t=1}^{|o_i|} A_i \cdot \log \pi_{\theta, i, t}, \quad \text{where } A_i = r_i - m. \quad (3)$$

Group Sequence Policy Optimization (**GSPO**) (Zheng et al., 2025) replaces the token-level probability ratio $\pi_{\theta, i, t}$ with a sequence-level one, denoted as $s_{\theta, i}$. Its objective then incorporates this ratio into a clipped surrogate function, closely resembling the PPO structure:

$$\mathcal{J}_{\text{GSPO}}(\theta) = \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \frac{1}{G} \sum_{i=1}^G \min \left[s_{\theta, i} \hat{A}_i, \text{clip}_{\epsilon}(s_{\theta, i}) \hat{A}_i \right], \quad \text{where } s_{\theta, i} = \left(\frac{\pi_{\theta}(o_i | q)}{\pi_{\theta_{\text{old}}}(o_i | q)} \right)^{\frac{1}{|o_i|}}. \quad (4)$$

The sequence-level ratio $s_{\theta, i}$ is defined as the geometric mean of the token-level probability ratios, which is implemented by $\exp\left(\frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \log \frac{\pi_{\theta, i, t}}{\pi_{\theta_{\text{old}}, i, t}}\right)$ in practice.

2.2 THE OVERCONFIDENCE ISSUE AND CURRENT SOLUTIONS

As aforementioned, GRPO and its variants often result in overconfident models (Leng et al., 2025; Bereket & Leskovec, 2025), which necessitates confidence calibration to ensure their reliability. The practical risks posed by such overconfidence have been well-studied (Melotti et al., 2022; Omar et al., 2024; Griot et al., 2025). In risk-sensitive applications such as disease diagnosis and auto-driving, confidence scores are key signals for introducing human auditing or fallback mechanisms. Overconfident models are unreliable (and inapplicable) in such applications because their mistakes are hard to predict. An LLM’s confidence is essentially its own prediction of how likely its answer is to be right, while calibration is the test of how reliable the prediction is — it checks whether the model’s stated confidence levels match its true success rate (Yoon et al., 2025b). Several methods have been developed to estimate this confidence. Gupta et al. (2024) uses the token-level probability of the answer to form a confidence score. Kadavath et al. (2022) employs other models to predict correctness. Xu et al. (2024) leverages labeled confidence scores datasets to fine-tune models. The methods in (Yoon et al., 2025b; Damani et al., 2025) elicit verbalized confidence, which prompts the model to articulate its own confidence estimate directly within its output.

Traditional confidence calibration approaches often operate at inference time by prompting the model to self-assess its understanding (Wen et al., 2024) or by introducing distractors (Chhikara, 2025). However, they often focus on scenarios of verbalized confidence, with unverified applicability to token-probability-based confidence. Recent works have sought to integrate calibration directly through regularization. For example, RLCR (Damani et al., 2025) designs a hybrid reward combining accuracy with a Brier calibration score. Taming LLMs (Leng et al., 2025) dynamically modifies reward values during PPO training based on recent performance. Similarly, CoDaPO (Zhou et al., 2025) modulates the advantage function with confidence and difficulty weights. Despite their contributions, these training-based methods often struggle to achieve a trade-off between confidence and accuracy, sometimes sacrificing performance or yielding suboptimal calibration.

3 CONFIDENCE-CALIBRATED GROUP SEQUENCE POLICY GRADIENT

To mitigate the model overconfidence issue while maintaining high reasoning performance, we propose a method, called Confidence-Calibration Group Sequence Policy Gradient (**C²GSPG**), that

introduces a novel objective function to explicitly align a model’s self-assessed confidence with its task performance. **Our method is motivated by a desideratum commonly used in confidence calibration (Grabinski et al., 2022; Wen et al., 2024; Yoon et al., 2025a): the probability of generated sequence should reflect the model confidence corresponding to the prompt/question.** Therefore, our method defines model confidence directly from the model’s sequence-level probabilities, rather than using verbalized confidence statements (Bereket & Leskovec, 2025). This is universally applicable and provides a more direct measure of the model’s internal certainty (Gupta et al., 2024). Formally, for the i -th sequence o_i corresponding to the prompt q , we compute the model confidence of o_i as

$$c_{\theta,i} := \pi_{\theta}(o_i|q)^{\frac{1}{|o_i|}} = \exp\left(\frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \log \pi_{\theta,i,t}\right). \quad (5)$$

Here, $\pi_{\theta}(o_i|q)^{\frac{1}{|o_i|}}$ denotes normalized sequence-level probability, which can be computed based on the token-level probabilities $\{\pi_{\theta,i,t}\}$. The normalization eliminates the impact of sequence length.

3.1 IMPLEMENTATION AND ANALYSIS IN BINARY REWARD SCENARIOS

In principle, the objective function of our method consists of a policy optimization term and a confidence calibration regularizer, both of which hinge on the confidence $c_{\theta,i}$. In the following content, we will introduce its implementation and analyze its rationality in theory. For simplification, we first implement and analyze our method in binary reward scenarios. For each prompt q and a group of generated sequences $\{o_i\}_{i=1}^G$, we apply a group sequence policy gradient loss with a binary cross-entropy regularizer to implement this objective, i.e.,

$$\mathcal{J}_{C^2\text{GSPG}}(\theta) = \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \frac{1}{G} \sum_{i=1}^G \left[\underbrace{\tilde{A}_i \cdot \log c_{\theta,i}}_{\text{Policy gradient}} + \beta \underbrace{[r_i \log c_{\theta,i} + (1-r_i) \log(1-c_{\theta,i})]}_{\text{Binary cross-entropy regularizer}} \right], \quad (6)$$

where $r_i \in \{0, 1\}$ is binary reward achieved by o_i . \tilde{A}_i is a normalized advantage defined as

$$\tilde{A}_i = \frac{r_i - m}{1 - c_{\theta_{\text{old}},i}} = \frac{r_i - \frac{1}{G} \sum_{j=1}^G r_j}{1 - c_{\theta_{\text{old}},i}}, \quad (7)$$

where $c_{\theta_{\text{old}},i}$ denotes the confidence of the old policy, and m is the average of the binary rewards in the group. For each sequence o_i , the policy gradient term $\tilde{A}_i \cdot \log c_{\theta,i}$ in (6) pursues the consistency between the model confidence and the advantage function — the high-confident sequence should achieve a high advantage. The cross-entropy regularizer $\beta[r_i \log c_{\theta,i} + (1-r_i) \log(1-c_{\theta,i})]$ penalizes the discrepancy between the reward and the model confidence of the sequence, whose significance is controlled by the hyperparameter $\beta > 0$. This regularizer pushes the model confidence towards 1 for high-reward sequences and towards 0 for low-reward sequences.

Remark. The design of \tilde{A}_i integrates group-level advantage with model confidence, thereby introducing confidence as a fine-grained modulation factor across sequences within the same group. When the answer is incorrect, we have $r_i - m \leq 0$, in which a smaller c_i leads to a larger \tilde{A}_i . Conversely, when the answer is correct and $r_i - m \geq 0$, a larger c_i results in a larger \tilde{A}_i . In other words, this modulation aligns with the objective of model calibration, encouraging the model to be more confident for truly advantageous sequences while reducing confidence for disadvantageous ones.

By defining our objective in this manner, we formally analyze its underlying optimization dynamics. In particular, according to the definition in (5), the gradient of $c_{\theta,i}$ at θ is

$$\nabla_{\theta} c_{\theta,i} = c_{\theta,i} \cdot \left(\frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \nabla_{\theta} \log \pi_{\theta,i,t} \right). \quad (8)$$

When computing the gradient of $\mathcal{J}_{C^2\text{GSPG}}(\theta)$ at θ , we apply the gradient chain rule and obtain

$$\begin{aligned} \nabla_{\theta} \mathcal{J}_{C^2\text{GSPG}}(\theta) &= \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \frac{1}{G} \sum_{i=1}^G \left(\tilde{A}_i + \beta \frac{r_i - c_{\theta,i}}{1 - c_{\theta,i}} \right) \left(\frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \nabla_{\theta} \log \pi_{\theta,i,t} \right) \\ &= \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \frac{1}{G} \sum_{i=1}^G \left(\frac{r_i - m}{1 - c_{\theta_{\text{old}},i}} + \beta \frac{r_i - c_{\theta,i}}{1 - c_{\theta,i}} \right) \left(\frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \nabla_{\theta} \log \pi_{\theta,i,t} \right). \end{aligned} \quad (9)$$

As shown in (9), the rationality and effectiveness of our proposed training objective hinge on the collaboration between the policy optimization term and the confidence-calibration regularizer, whose directions are determined by $r_i - m$ and $r_i - c_{\theta,i}$, respectively. As shown in the following proposition, these two terms work in synergy for binary rewards — the model’s confidence is adjusted in a manner that is consistent with the policy update.

Proposition 3.1. *Given binary rewards $r_i \in \{0, 1\}$, the group mean reward $m \in (0, 1)$ and the model’s confidence $c_{\theta,i} \in (0, 1)$, the policy optimization direction $r_i - m$ and the regularization direction $r_i - c_{\theta,i}$ are consistent, that is, they always share the same sign.*

The proof of this proposition is straightforward. As shown in Table 1, for a correct response ($r_i = 1$), the policy optimization direction $r_i - m$ and the regularization direction $r_i - c_{\theta,i}$ are both nonnegative, while for an incorrect response ($r_i = 0$), they are both non-positive. Such consistency ensures that the model becomes more confident in correct responses and less confident in incorrect ones.

To further highlight the effectiveness of C²GSPG, given a sequence o_i , we compare it with several representative methods (including GRPO (Shao et al., 2024), GPG (Chu et al., 2025), AR-Lopti (Yang et al., 2025b), and GSPO (Zheng et al., 2025)) on their gradient formulations. Table 2 shows that C²GSPG offers the following two advantages.¹

1) The elimination of token-level bias.

Proposition 4.2 in (Yang et al., 2025b) shows that the gradient of GRPO suffers the over-dominance of low-probability tokens. AR-Lopti (Yang et al., 2025b) introduces token-specific adaptive weights to mitigate this issue. The sequence-based policy optimization methods, including GPG (Chu et al., 2025), GSPO (Zheng et al., 2025), and our C²GSPG, can alleviate the token-level bias by computing a single policy gradient signal for an entire sequence. This setting treats different tokens evenly, which promotes a more stable and uniform training process.

2) The mitigation of sequence-level bias.

Our method provides a more robust mechanism for leveraging model confidence. As shown in Table 2, the advantage term (i.e., $r_i - m$) of GSPO is modulated by $c_{\theta,i}/(c_{\theta_{old},i}\sigma)$. It means that when the current policy π_θ has a high uncertainty on a question q (i.e., σ is large) and generates a low-confidence sequence o_i , GSPO tends to weaken the significance of q and o_i in the gradient. As a result, GSPO (Zheng et al., 2025) relies more on those high-confidence sequences during training and thus inevitably suffers from the overconfidence issue. On the contrary, C²GSPG modulates the advantage term by $1/(1 - c_{\theta_{old},i})$. This weight is in the range $[1, \infty)$, avoiding the gradient diminishing of low-confidence sequences. Concurrently, we omit the variance-based normalization because this technique has recently been shown to produce uncalibrated confidence estimates (Bereket & Leskovec, 2025).

3.2 MITIGATING GRADIENT CONFLICTS IN NON-BINARY REWARD SCENARIOS

In Proposition 3.1, we establish that for binary reward scenarios, the policy gradient term and the confidence-based regularizer always share the same sign. However, this synergy does not inherently extend to scenarios with non-binary rewards (e.g., the rewards in the K&K dataset (Xie et al., 2024)

¹When formulating the gradients, we ignore the KL-divergence term and the clipping function. Such a simplification does not impact the correctness of our analysis. For AR-Lopti, the hyperparameter $\eta \in [0, 1]$. For GSPO, the $s_{\theta,i}$ in (4) is equal to $c_{\theta,i}/c_{\theta_{old},i}$.

Table 1: The directions of different terms in the gradient in (9).

Weight	Correct o_i	Incorrect o_i
$r_i - m$	≥ 0	≤ 0
$r_i - c_{\theta,i}$	≥ 0	≤ 0

Table 2: A comparison of the gradient per sequence. We denote $\nabla_\theta \log \pi_{\theta,i,t}$ as $g_{\theta,i,t}$ for short. The methods having **red** terms treat tokens of a sequence differently, while the others treat all tokens evenly. The **blue** terms show the impact of model confidence on advantage.

Method	Gradient per sequence
GRPO	$\frac{r_i - m}{ o_i \sigma} \sum_{t=1}^{ o_i } \frac{\pi_{\theta,i,t}}{\pi_{\theta_{old},i,t}} g_{\theta,i,t}$
AR-Lopti	$\frac{r_i - m}{ o_i \sigma} \sum_{t=1}^{ o_i } [\eta \pi_{\theta_{old},i,t} + (1 - \eta)] \frac{\pi_{\theta,i,t}}{\pi_{\theta_{old},i,t}} g_{\theta,i,t}$
GPG	$\frac{1}{\sum_{j=1}^C o_j } (r_i - m) \left(\sum_{t=1}^{ o_i } g_{\theta,i,t} \right)$
GSPO	$\frac{c_{\theta,i}(r_i - m)}{c_{\theta_{old},i}\sigma} \left(\frac{1}{ o_i } \sum_{t=1}^{ o_i } g_{\theta,i,t} \right)$
C ² GSPG	$\left(\frac{r_i - m}{1 - c_{\theta_{old},i}} + \frac{\beta(r_i - c_{\theta,i})}{1 - c_{\theta,i}} \right) \left(\frac{1}{ o_i } \sum_{t=1}^{ o_i } g_{\theta,i,t} \right)$

are $\{-3, -1, -0.5, 3\}$). In particular, a gradient conflict can arise when a reward r_i falls between the mean reward m and the model confidence $c_{\theta,i}$. To mitigate this risk, we apply a nonlinear normalization method to rewards and an adaptive clipping strategy to the regularizer weight β , respectively, reformulating the objective function of C²GSPG as

$$\mathcal{J}_{\text{C}^2\text{GSPG}}(\theta) = \mathbb{E}_{\{o_i\} \sim \pi_{\theta, \text{old}}} \frac{1}{G} \sum_{i=1}^G \left[\tilde{A}_i \cdot \log c_{\theta,i} + \mathbb{I}_i(\beta) \cdot [\hat{r}_i \log c_{\theta,i} + (1 - \hat{r}_i) \log(1 - c_{\theta,i})] \right], \quad (10)$$

$$\text{where } \hat{r}_i = \sigma(\alpha \cdot r_i) = \frac{1}{1 + \exp(-\alpha \cdot r_i)}, \quad \mathbb{I}_i(\beta) = \begin{cases} \beta, & \text{if } \text{sign}(\hat{r}_i - \hat{m}) = \text{sign}(\hat{r}_i - c_{\theta,i}), \\ 0, & \text{otherwise.} \end{cases}$$

Here, \hat{r}_i is a sigmoid function that makes a nonlinear normalization on the original rewards. This maps any arbitrary reward to a standardized range of $[0, 1]$, aligning its scale with that of the confidence score $c_{\theta,i}$. The α controls the steepness of the normalization. Accordingly, the advantage is computed based on the normalized reward, i.e., $\tilde{A}_i = (\hat{r}_i - \hat{m}) / (1 - c_{\theta, \text{old}, i})$, where the mean reward is $\hat{m} = \sum_{i=1}^G \hat{r}_i$. As illustrated in Figure 2, the sigmoid function is a monotonically increasing function, which maintains the order of the rewards. Moreover, by pushing positive and negative rewards towards 1 and 0, respectively, it amplifies the gap between positive and negative feedback and thus reduces the risk of gradient conflict. In addition, for the sequence whose normalized reward \hat{r}_i is still between \hat{m} and $c_{\theta,i}$, the adaptive clipping function $\mathbb{I}_i(\beta) = 0$, ensuring that a conflicting confidence adjustment never counteracts a valid policy signal. In Appendix A.2, we show the scheme of C²GSPG in detail.

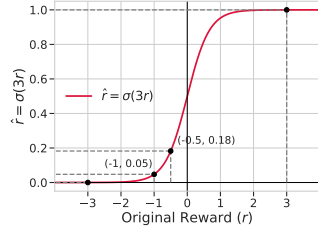


Figure 2: The normalization of the rewards in the K&K dataset (Xie et al., 2024).

4 EXPERIMENTS

To validate the effectiveness of C²GSPG, we conduct experiments on logical and mathematical reasoning tasks. In each task, we compare C²GSPG with state-of-the-art methods, including GRPO (Shao et al., 2024), GPG (Chu et al., 2025), GSPO (Zheng et al., 2025), and AR-Lopti (Yang et al., 2025b). **Representative experimental results are shown below, and more implementation details and results are given in the Appendix.**

4.1 IMPLEMENTATION DETAILS

For the logical reasoning task, we employ the K&K dataset (Xie et al., 2024), stratified by character count (3 to 7) to represent varying difficulty. Following the methodology of (Yang et al., 2025b), we select the Qwen2.5-3B-Instruct model² as our backbone and train it on a composite dataset encompassing all difficulty levels. Our training process is guided by a reward function adapted from Logic-RL (Xie et al., 2025), which combines structural and correctness signals. The raw discrete reward signal of $\{-3, -1, -0.5, 3\}$ is transformed through the proposed sigmoid mapping: boundary scores -3 and 3 are mapped to 0 and 1, while intermediate values are converted via (10) (with $\alpha = 3$), yielding a final reward set of $\{0, 0.047426, 0.182426, 1\}$. The hyperparameter $\beta = 0.03$, and we pass it through the proposed clipping function $\mathbb{I}(\cdot)$.

For the mathematical reasoning task, we utilize the Qwen2.5-Math-1.5B model³, which we train on the DeepScaleR-Uniform dataset (Shi et al., 2025), a curated subset of the DeepScaleR benchmark (Luo et al., 2025). Our training methodology employs a straightforward binary reward mechanism: a reward of $+1$ is assigned if the model’s final answer is correct, and 0 otherwise. In this case, we set $\beta \in \{0.01, 0.1, 0.5\}$. To assess generalization, we evaluate performance across a diverse suite of benchmarks, including GSM8K (Cobbe et al., 2021), Olympiad (He et al., 2024), Minerva (Lewkowycz et al., 2022), MATH 500 (Lightman et al., 2024), AMC2023, and AIME2024.

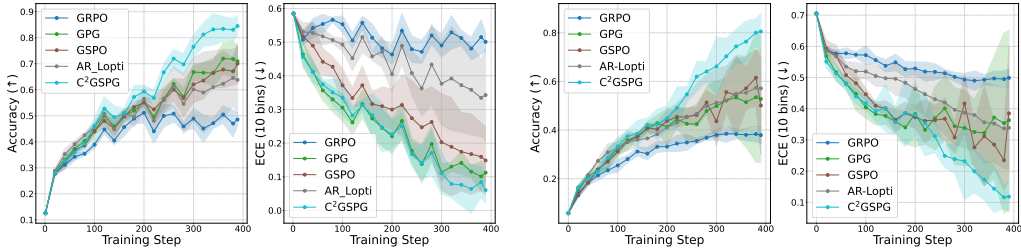
For each trained model π_{θ^*} , the primary performance indicator is **accuracy**, calculated as the fraction of correctly answered questions. To further assess the reliability of the model’s confidence

²The model is available at: <https://huggingface.co/Qwen/Qwen2.5-3B-Instruct>

³The model is available at: <https://huggingface.co/Qwen/Qwen2.5-Math-1.5B>

Table 3: The comparison for various methods on the K&K dataset. The row of ‘‘Base’’ shows the performance of Qwen2.5-3B-Instruct without post-training. In each column, the best result is **bold**.

Method	Accuracy (%) by # Characters					Overall Performance		
	3	4	5	6	7	Acc (↑)	BS (↓)	ECE (↓)
Base	12.00	13.00	2.00	2.00	0.00	5.80	0.554	0.705
GRPO	58.67±4.20	44.93±3.99	34.53±2.69	31.87±2.92	20.53±3.01	38.11±2.68	0.405±0.018	0.497±0.023
GPG	69.53±17.58	64.80±25.28	49.47±28.39	43.20±24.87	35.93±21.37	52.59±22.67	0.363±0.230	0.363±0.242
GSPO	71.13±12.63	66.40±15.61	55.40±11.95	48.20±9.89	40.07±11.04	56.24±11.67	0.291±0.102	0.302±0.106
AR-Lopti	73.20±9.45	64.93±16.05	54.13±16.13	49.47±17.04	41.33±19.27	56.61±15.42	0.282±0.101	0.343±0.128
C²GSPG	86.00±5.53	88.00±4.99	78.60±6.26	73.27±10.53	69.00±9.45	78.97±7.19	0.136±0.046	0.126±0.053



(a) The model performance on the training set

(b) The model performance on the testing set

Figure 3: Training dynamics on the K&K dataset. The right shows the evolution of Accuracy and ECE on the training set. The left shows the same metrics on the test set.

scores, we measure its calibration by computing the **Brier Score (BS)** (Glenn et al., 1950) and the **Expected Calibration Error (ECE)** (Naeini et al., 2015), which are defined as

$$BS = \frac{1}{N} \sum_{i=1}^N (c_{\theta^*,i} - r_i)^2, \quad ECE = \sum_{m=1}^M \frac{|B_m|}{N} |\text{acc}(B_m) - \text{conf}(B_m)|. \quad (11)$$

According to the definition, BS quantifies the mean squared difference between the predicted confidence $c_{\theta^*,i}$ and the actual outcome r_i over all N samples. ECE measures the discrepancy between confidence and accuracy by grouping predictions into M bins. It is the weighted average of the absolute difference between the average accuracy and average confidence in each bin B_m . In our experiments, we set the number of bins to $M = 10$. For each evaluation metric, we run each method in five trials and record the mean and standard deviation of its performance.

4.2 LOGICAL REASONING

Table 3 summarizes the performance of various methods on the K&K dataset, with a detailed breakdown by difficulty level (number of characters) and overall calibration metrics. In terms of overall performance, C²GSPG achieves an average accuracy that is over 22% higher than the strongest baseline, AR-Lopti, and the accuracy improvement is complemented by lower BS and ECE values. It means that C²GSPG not only makes more accurate predictions but also produces more reliable confidence estimates. Furthermore, while all methods exhibit performance degradation as puzzle difficulty increases, C²GSPG shows superior robustness. For instance, its performance lead over AR-Lopti widens from approximately 13% on 3-character puzzles to nearly 28% on the most challenging 7-character puzzles. This highlights the enhanced capability of our method to handle more complex reasoning tasks. Finally, the consistently small standard deviations across different runs underscore the stability and reliability of our approach.

In addition, we provide a quantitative and qualitative visualization analysis of different methods. In Figure 3, we illustrate the evolution of Accuracy and ECE on both the training and test sets. These plots show that C²GSPG consistently achieves higher accuracy and lower ECE across both training and test sets, demonstrating its strong data fitting ability and generalization power.

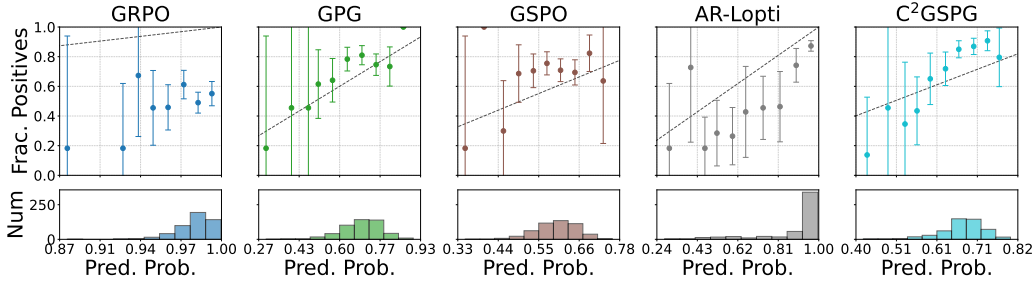


Figure 4: Calibration performance of various methods on the K&K dataset (test set). The top plot shows the Fraction of Positives against the model’s predicted confidence, with the dashed line representing perfect calibration. The confidence histogram (bottom) shows the distribution of predicted confidences. Results are generated using a sampling strategy with a temperature of 1.0 and allowing the entire probability distribution to be considered.

Table 4: Ablation studies on the K&K dataset. We compare different variants of C²GSPG with an adaptively-clipped or fixed regularization weight and a linear or sigmoid-based reward normalization. Here, the linear reward normalization is achieved by $\hat{r}_i = (r_i - r_{\min}) / (r_{\max} - r_{\min})$.

Method	Accuracy (%) by # Characters					Overall Performance				
	$\mathbb{I}(\cdot)$	$\sigma(\cdot)$	3	4	5	6	7	Acc (\uparrow)	BS (\downarrow)	ECE (\downarrow)
×	Linear		77.93 \pm 3.64	75.47 \pm 7.95	63.13 \pm 12.16	53.60 \pm 12.13	44.93 \pm 12.36	63.01 \pm 9.36	0.229 \pm 0.028	0.226 \pm 0.040
✓	Linear		79.33 \pm 5.67	79.87 \pm 7.20	70.07 \pm 6.69	64.00 \pm 9.15	57.20 \pm 10.33	70.09 \pm 7.60	0.169 \pm 0.059	0.171 \pm 0.048
×	Sigmoid		78.93 \pm 6.09	79.13 \pm 11.17	66.33 \pm 15.14	60.33 \pm 18.08	56.53 \pm 19.05	68.25 \pm 13.86	0.216 \pm 0.110	0.208 \pm 0.112
✓	Sigmoid		86.00\pm5.53	88.00\pm4.99	78.60\pm6.26	73.27\pm10.53	69.00\pm9.45	78.97\pm7.19	0.136\pm0.046	0.126\pm0.053

To demonstrate that C²GSPG helps calibrate model confidence, we compare it with baseline methods on their calibration curves and the histograms of confidence scores. The top plots in Figure 4 show that the calibration curve of C²GSPG aligns closely to the perfect-calibration diagonal. In contrast, the baseline methods either result in a calibration curve falling below the diagonal (e.g., GRPO and AR-Lopti) or exhibit undesired high variance for their high-confidence sequences (e.g., GSPO), indicating a tendency towards overconfidence. In addition, some baselines fail to exhibit clear monotonically increasing curves, suggesting they do not effectively differentiate between levels of confidence when answering questions. The histograms of various methods (the bottom plots in Figure 4) further show that C²GSPG produces a more evenly spread confidence distribution, while GRPO and AR-Lopti display a highly concentrated one with predictions clustered at the extremes.

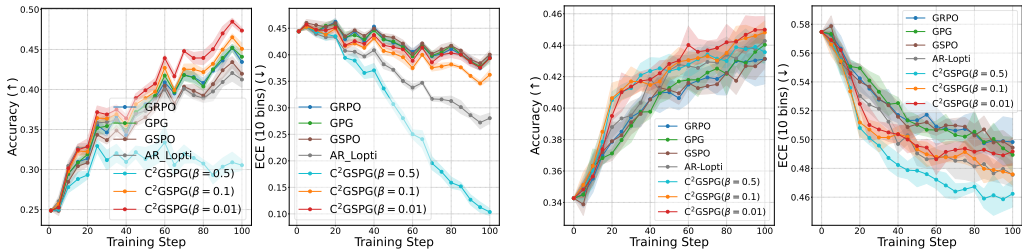
Ablation Studies. To validate the rationality of our method, we conduct a series of ablation studies in Table 4, quantitatively analyze the impacts of adaptive regularizer clipping and sigmoid-based reward normalization on model performance. For the K&K dataset, given a set of sequences and their rewards, we apply C²GSPG under different configurations. The results in Table 4 demonstrate that both the adaptive clipping mechanism and the sigmoid-based normalization help improve model performance consistently, leading to higher accuracy and better confidence calibration performance. These results demonstrate the necessity of mitigating gradient conflict in non-binary reward scenarios. In addition, the results in Tables 3 and 4 show that the simplified C²GSPG, with fixed regularization weight and linear reward normalization, can still outperform all four baseline methods. This phenomenon indicates the effectiveness of our group sequence policy gradient framework.

4.3 MATHEMATICAL REASONING

Table 5 presents the results on the mathematical reasoning task. We compare C²GSPG with varying values of β against the baselines. Our results highlight a tunable trade-off between accuracy and calibration, governed by the regularization strength β . With a lower β , our approach surpasses baselines in terms of accuracy. As β increases, the focus shifts towards calibration, resulting in superior BS and ECE at the cost of a modest dip in accuracy. The calibration results in Figure 1(a) further verify our above claim. The calibration curve for C²GSPG is distinguished by its close alignment

Table 5: The comparison for various methods on mathematical reasoning benchmarks. The subscript in C^2 GSPG indicates the value of β . The row of “Base” shows the performance of Qwen2.5-Math-1.5B without post-training. In each column, the best result is **bold**.

Method	Testing Data Accuracy (%)					Overall Performance			
	GSM8K	MATH500	AMC2023	AIME2024	Olympiad Minerva	Acc (\uparrow)	BS (\downarrow)	ECE (\downarrow)	
Base	67.63	58.00	44.69	6.25	19.11	9.93	34.27	0.555	0.575
GRPO	79.57 \pm 0.62	69.92 \pm 0.48	54.67 \pm 2.67	13.06 \pm 3.02	23.47 \pm 0.45	15.88 \pm 0.30	42.76 \pm 0.96	0.491 \pm 0.008	0.501 \pm 0.010
GPG	79.85 \pm 0.51	69.15 \pm 0.93	56.81 \pm 2.99	13.94 \pm 2.50	22.77 \pm 0.32	15.64 \pm 1.45	43.03 \pm 0.98	0.489 \pm 0.010	0.498 \pm 0.011
GSPO	80.30 \pm 0.34	69.87 \pm 0.59	52.60 \pm 2.46	13.64 \pm 1.49	23.18 \pm 0.41	16.50 \pm 0.21	42.68 \pm 0.43	0.489 \pm 0.005	0.498 \pm 0.005
AR-Lopti	79.91 \pm 0.58	69.76 \pm 0.47	57.08\pm1.76	15.08 \pm 2.39	24.07\pm0.32	17.08 \pm 0.66	43.83 \pm 0.58	0.468 \pm 0.005	0.478 \pm 0.006
C^2 GSPG _{0.01}	82.72\pm0.15	72.15\pm0.75	54.25 \pm 2.61	16.33\pm2.13	23.86 \pm 0.41	18.36 \pm 0.72	44.61\pm0.62	0.484 \pm 0.005	0.491 \pm 0.006
C^2 GSPG _{0.1}	82.66 \pm 0.51	70.97 \pm 0.26	54.37 \pm 2.13	15.39 \pm 1.83	23.56 \pm 0.41	18.58\pm0.75	44.25 \pm 0.51	0.473 \pm 0.004	0.481 \pm 0.005
C^2 GSPG _{0.5}	82.29 \pm 0.37	71.07 \pm 0.53	52.88 \pm 3.41	13.75 \pm 2.01	23.26 \pm 0.23	17.38 \pm 1.02	43.44 \pm 0.67	0.463\pm0.007	0.464\pm0.007



(a) The model performance on the training set

(b) The model performance on the testing set

Figure 5: Training dynamics on the mathematical dataset. The right shows the evolution of Accuracy and ECE on the training set. The left shows the same metrics on the test set.

with the ideal diagonal and its consistent positive slope. In contrast, all other baseline methods suffer from severe overconfidence, with their calibration curves falling far below the diagonal. Figure 5 visualizes the training dynamics of different methods and the impact of β on our method.

It should be noted that the trade-off shown in the results is not in conflict with Proposition 3.1. In binary reward scenarios, the regularizer and the policy optimization term have the same gradient direction indeed. In other words, the regularizer implicitly increases the learning rate according to the consistency between the confidence score and the reward of each sequence. As a result, an improperly large β may lead to an over-aggressive step size during training and thus result in performance degradation.

4.4 FURTHER ANALYSIS

The results in Tables 3 and 5 indicate that our method achieves significant improvements in logical reasoning but incremental gains in mathematical reasoning. In addition, when changing β , the reasoning accuracy and confidence calibration metrics (BS and ECE) show a trade-off on the math datasets, whereas improvements in these metrics are consistent across the K&K dataset, as shown in Table 10 in the Appendix. Note that these two phenomena are common (Zhang et al., 2025; Lacombe et al., 2025; Yuan et al., 2025), whose reasons are the difficulties of the reasoning tasks and the capabilities of the pre-trained LLMs w.r.t. the tasks.

- When the pre-trained LLM is strong while the task is relatively easy (e.g., K&K), the LLM possesses sufficient latent capacity for the task. Thus, fine-tuning with different methods can yield substantial performance gains. Notably, since K&K does not appear in the pre-training corpora of mainstream LLMs, these performance improvements primarily stem from genuine learning rather than data memorization. Consequently, the performance improvement of our method compared to other baselines further validates its effectiveness.

- When the pre-trained LLM is not strong enough while the task is challenging (e.g., the mathematical datasets), the work in (Yue et al., 2025) points out that although RL methods can improve sample efficiency, they remain unable to exceed the upper bound of their capabilities, thus resulting in limited performance gains. In such a situation, an overconfident model is encouraged by RL methods (including ours) because overconfident guessing may yield higher rewards, whereas an honest response often leads to failure. To maximize rewards, the model resorts to strategic guessing, leading to improved accuracy but degraded confidence calibration.

To address the above issue, we will further explore how to improve the calibration of models whose capabilities are currently insufficient for handling challenging tasks.

5 CONCLUSION

In this work, we propose C²GSPG, a confidence-calibrated group sequence policy gradient method designed to suppress the overconfidence issue in reinforcement learning-based reasoning models. Within the group sequence policy gradient framework, C²GSPG introduces a confidence-modulated advantage and confidence calibration regularizer, which effectively eliminates token-level bias and enforces consistency between model confidence and rewards. Experimental results on logical and mathematical reasoning tasks demonstrate that C²GSPG significantly improves both reasoning accuracy and confidence calibration, outperforming existing state-of-the-art methods.

Limitations and Future Work. On the mathematical reasoning task, we observe that the gap in confidence calibration between the training and test sets remains relatively large, which may stem from the distribution discrepancy between the two datasets. Additionally, the model performance is significantly impacted by the choice of β . Addressing these issues, we plan to explore methods to enhance out-of-distribution robustness and calibration, as well as develop more effective regularization strategies to suppress overconfidence in erroneous generations.

ETHICS STATEMENT

The datasets used in this paper are all publicly available and do not involve any ethical issues.

REPRODUCIBILITY STATEMENT

We are committed to ensuring the reproducibility of our research. The source code is provided in the supplementary materials, which can be used to reproduce the main results presented in this paper.

REFERENCES

- Michael Bereket and Jure Leskovec. Uncalibrated reasoning: GRPO induces overconfidence for stochastic outcomes. In *LLM for Scientific Discovery: Reasoning, Assistance, and Collaboration*, 2025.
- Prateek Chhikara. Mind the confidence gap: Overconfidence, calibration, and distractor effects in large language models. *arXiv preprint arXiv:2502.11028*, 2025.
- Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. GPG: A simple and strong reinforcement learning baseline for model reasoning. *arXiv preprint arXiv:2504.02546*, 2025.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

- 540 Mehul Damani, Isha Puri, Stewart Slocum, Idan Shenfeld, Leshem Choshen, Yoon Kim, and Jacob
541 Andreas. Beyond binary rewards: Training lms to reason about their uncertainty. *arXiv preprint*
542 *arXiv:2507.16806*, 2025.
- 543
544 W Brier Glenn et al. Verification of forecasts expressed in terms of probability. *Monthly weather*
545 *review*, 78(1):1–3, 1950.
- 546 Julia Grabinski, Paul Gavrikov, Janis Keuper, and Margret Keuper. Robust models are less over-
547 confident. In *Proceedings of the 36th International Conference on Neural Information Processing*
548 *Systems*, pp. 39059–39075, 2022.
- 549
550 Maxime Griot, Coralie Hemptinne, Jean Vanderdonckt, and Demet Yuksel. Large language models
551 lack essential metacognition for reliable medical reasoning. *Nature communications*, 16(1):642,
552 2025.
- 553
554 Neha Gupta, Harikrishna Narasimhan, Wittawat Jitkrittum, Ankit Singh Rawat, Aditya Krishna
555 Menon, and Sanjiv Kumar. Language model cascades: Token-level uncertainty and beyond. In
556 *International Conference on Learning Representations*, 2024.
- 557
558 Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han,
559 Yujie Huang, Yuxiang Zhang, et al. OlympiadBench: A challenging benchmark for promoting
560 AGI with Olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd*
561 *Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp.
3828–3850, 2024.
- 562
563 Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec
564 Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. OpenAI o1 system card. *arXiv*
565 *preprint arXiv:2412.16720*, 2024.
- 566
567 Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez,
568 Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language mod-
els (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.
- 569
570 Romain Lacombe, Kerrie Wu, and Eddie Dilworth. Don’t think twice! over-reasoning impairs
571 confidence calibration. *arXiv preprint arXiv:2508.15050*, 2025.
- 572
573 Jixuan Leng, Chengsong Huang, Banghua Zhu, and Jiabin Huang. Taming overconfidence in LLMs:
574 Reward calibration in RLHF. In *International Conference on Learning Representations*, 2025.
- 575
576 Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ra-
577 masesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative
578 reasoning problems with language models. In *Proceedings of the 36th International Conference*
579 *on Neural Information Processing Systems*, pp. 3843–3857, 2022.
- 580
581 Chen Li, Nazhou Liu, and Kai Yang. Adaptive group policy optimization: Towards stable training
582 and token-efficient reasoning. *arXiv preprint arXiv:2503.15952*, 2025.
- 583
584 Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan
585 Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let’s verify step by step. In *International*
586 *Conference on Learning Representations*, 2024.
- 587
588 Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y Tang, Manan Roongta, Colin Cai,
589 Jeffrey Luo, Tianjun Zhang, Li Erran Li, et al. Deepscaler: Surpassing o1-preview with a 1.5B
590 model by scaling RL. *Notion Blog*, 2025.
- 591
592 Gledson Melotti, Cristiano Premebida, Jordan J Bird, Diego R Faria, and Nuno Goncalves. Reducing
593 overconfidence predictions in autonomous driving perception. *IEEE Access*, 10:54805–54821,
2022.
- 594
595 Mahdi Pakdaman Naeni, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated proba-
bilities using bayesian binning. In *Proceedings of the AAAI conference on artificial intelligence*,
volume 29, 2015.

- 594 Mahmud Omar, Benjamin S Glicksberg, Girish N Nadkarni, and Eyal Klang. Overconfident ai?
595 benchmarking llm self-assessment in clinical scenarios. *medRxiv*, 2024.
596
- 597 Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea
598 Finn. Direct preference optimization: your language model is secretly a reward model. In *Pro-*
599 *ceedings of the 37th International Conference on Neural Information Processing Systems*, pp.
600 53728–53741, 2023.
- 601 John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy
602 optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
603
- 604 Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang,
605 Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathemati-
606 cal reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- 607 Taiwei Shi, Yiyang Wu, Linxin Song, Tianyi Zhou, and Jieyu Zhao. Efficient reinforcement finetun-
608 ing via adaptive curriculum learning. *arXiv preprint arXiv:2504.05520*, 2025.
609
- 610 Bingbing Wen, Chenjun Xu, HAN Bin, Robert Wolfe, Lucy Lu Wang, and Bill Howe. Mitigat-
611 ing overconfidence in large language models: A behavioral lens on confidence estimation and
612 calibration. In *NeurIPS 2024 Workshop on Behavioral Machine Learning*, volume 1, 2024.
- 613 Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih
614 Ghazi, and Ravi Kumar. On memorization of large language models in logical reasoning. In *The*
615 *4th Workshop on Mathematical Reasoning and AI at NeurIPS*, 2024.
616
- 617 Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu,
618 Zhirong Wu, and Chong Luo. Logic-RL: Unleashing LLM reasoning with rule-based reinforc-
619 ement learning. *arXiv preprint arXiv:2502.14768*, 2025.
- 620 Tianyang Xu, Shujin Wu, Shizhe Diao, Xiaoze Liu, Xingyao Wang, Yangyi Chen, and Jing Gao.
621 SaySelf: Teaching LLMs to express confidence with self-reflective rationales. In *Proceedings*
622 *of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 5985–5998,
623 2024.
- 624 An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu,
625 Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint*
626 *arXiv:2505.09388*, 2025a.
627
- 628 Zhihe Yang, Xufang Luo, Zilong Wang, Dongqi Han, Zhiyuan He, Dongsheng Li, and Yun-
629 jian Xu. Do not let low-probability tokens over-dominate in RL for LLMs. *arXiv preprint*
630 *arXiv:2505.12929*, 2025b.
- 631 Dongkeun Yoon, Seungone Kim, Sohee Yang, Sunkyoung Kim, Soyeon Kim, Yongil Kim, Eunbi
632 Choi, Yireun Kim, and Minjoon Seo. Reasoning models better express their confidence. *arXiv*
633 *preprint arXiv:2505.14489*, 2025a.
634
- 635 Dongkeun Yoon, Seungone Kim, Sohee Yang, Sunkyoung Kim, Soyeon Kim, Yongil Kim, Eunbi
636 Choi, Yireun Kim, and Minjoon Seo. Reasoning models better express their confidence. *arXiv*
637 *preprint arXiv:2505.14489*, 2025b.
- 638 Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian
639 Fan, Gaohong Liu, Lingjun Liu, et al. DAPO: An open-source LLM reinforcement learning
640 system at scale. *arXiv preprint arXiv:2503.14476*, 2025.
- 641 Danlong Yuan, Tian Xie, Shaohan Huang, Zhuocheng Gong, Huishuai Zhang, Chong Luo, Furu
642 Wei, and Dongyan Zhao. Efficient rl training for reasoning models via length-aware optimization.
643 *arXiv preprint arXiv:2505.12284*, 2025.
644
- 645 Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao
646 Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond
647 the base model? In *2nd AI for Math Workshop @ ICML 2025*, 2025. URL <https://openreview.net/forum?id=upehLVgqlb>.

648 Qingcheng Zeng, Weihao Xuan, Leyang Cui, and Rob Voigt. Thinking out loud: Do reasoning
649 models know when they're right? *arXiv preprint arXiv:2504.06564*, 2025.
650

651 Junjie Zhang, Guozheng Ma, Shunyu Liu, Haoyu Wang, Jiaying Huang, Ting-En Lin, Fei Huang,
652 Yongbin Li, and Dacheng Tao. A simple” motivation” can enhance reinforcement finetuning of
653 large reasoning models. In *NeurIPS 2025 Workshop on Efficient Reasoning*, 2025.

654 Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang,
655 Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint*
656 *arXiv:2507.18071*, 2025.
657

658 Zhanke Zhou, Xiangyu Lu, Chentao Cao, Brando Miranda, Tongliang Liu, Bo Han, and Sanmi
659 Koyejo. CoDaPo: Confidence and difficulty-adaptive policy optimization for post-training lan-
660 guage models. In *2nd AI for Math Workshop at ICML*, 2025.
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701

APPENDIX

A DETAILS OF OUR METHOD

A.1 THE GRADIENT DERIVATION OF OUR METHOD

We will give a detailed derivation of our C²GSPG method. First, we have

$$\nabla_{\theta} c_{\theta,i} = c_{\theta,i} \cdot \left(\frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \nabla_{\theta} \log \pi_{\theta,i,t} \right), \text{ and } \nabla_{\theta} \log c_{\theta,i} = \left(\frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \nabla_{\theta} \log \pi_{\theta,i,t} \right). \quad (12)$$

Accordingly, we have

$$\begin{aligned} & \nabla_{\theta} \mathcal{J}_{\text{C}^2\text{GSPG}}(\theta) \\ &= \nabla_{\theta} \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^G \left(\tilde{A}_i \log(c_{\theta,i}) + \beta (r_i \log(c_{\theta,i}) + (1 - r_i) \log(1 - c_{\theta,i})) \right) \right] \\ &= \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^G \left(\tilde{A}_i \nabla_{\theta} \log(c_{\theta,i}) + \beta r_i \nabla_{\theta} \log(c_{\theta,i}) + \beta (1 - r_i) \nabla_{\theta} \log(1 - c_{\theta,i}) \right) \right] \\ &= \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^G \left((\tilde{A}_i + \beta r_i) \nabla_{\theta} \log(c_{\theta,i}) + \beta (1 - r_i) \left(\frac{-\nabla_{\theta} c_{\theta,i}}{1 - c_{\theta,i}} \right) \right) \right] \\ &= \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^G \left((\tilde{A}_i + \beta r_i) \nabla_{\theta} \log(c_{\theta,i}) - \beta (1 - r_i) \frac{c_{\theta,i} \cdot \nabla_{\theta} \log(c_{\theta,i})}{1 - c_{\theta,i}} \right) \right] \\ &= \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^G \left(\tilde{A}_i + \beta r_i - \beta (1 - r_i) \frac{c_{\theta,i}}{1 - c_{\theta,i}} \right) \nabla_{\theta} \log(c_{\theta,i}) \right] \\ &= \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^G \left(\tilde{A}_i + \beta \frac{r_i(1 - c_{\theta,i}) - (1 - r_i)c_{\theta,i}}{1 - c_{\theta,i}} \right) \nabla_{\theta} \log(c_{\theta,i}) \right] \\ &= \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^G \left(\tilde{A}_i + \beta \frac{r_i - c_{\theta,i}}{1 - c_{\theta,i}} \right) \nabla_{\theta} \log(c_{\theta,i}) \right] \\ &= \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{G} \sum_{i=1}^G \left(\frac{r_i - m}{1 - c_{\theta_{\text{old},i}}} + \beta \frac{r_i - c_{\theta,i}}{1 - c_{\theta,i}} \right) \left(\frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \nabla_{\theta} \log \pi_{\theta,i,t} \right) \right]. \end{aligned} \quad (13)$$

In our method, the reason that we use the binary cross-entropy (BCE) based regularizer rather than mean squared error (MSE) for calibrating model confidence is shown below. The objective function with MSE loss is:

$$\mathcal{J}_{\text{C}^2\text{GSPG}}^{\text{MSE}}(\theta) = \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \frac{1}{G} \sum_{i=1}^G \left[\tilde{A}_i \cdot \log c_{\theta,i} - \beta (r_i - c_{\theta,i})^2 \right], \quad (14)$$

The gradient of it is:

$$\begin{aligned} & \nabla_{\theta} \mathcal{J}_{\text{C}^2\text{GSPG}}^{\text{MSE}}(\theta) \\ &= \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \frac{1}{G} \sum_{i=1}^G \left(\tilde{A}_i - 2\beta \cdot c_{\theta,i} (c_{\theta,i} - r_i) \right) \left(\frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \nabla_{\theta} \log \pi_{\theta,i,t} \right) \\ &= \mathbb{E}_{\{o_i\} \sim \pi_{\theta_{\text{old}}}} \frac{1}{G} \sum_{i=1}^G \left(\frac{r_i - m}{1 - c_{\theta_{\text{old},i}}} - 2\beta \cdot c_{\theta,i} (c_{\theta,i} - r_i) \right) \left(\frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \nabla_{\theta} \log \pi_{\theta,i,t} \right). \end{aligned} \quad (15)$$

From (15), we see that the MSE regularizer scales the gradient directly with $c_{\theta,i}(r_i - c_{\theta,i})$. For sequences with low predicted confidence, this results in a small gradient, producing a weak corrective signal. In contrast, as mentioned in Section 3.1, BCE introduces a $1/(1 - c_{\theta,i})$ modulation, which avoid this problem.

A.2 EXPERIMENTAL DETAILS

All experiments are performed using the VERL framework⁴ on NVIDIA A100 GPUs. The whole training algorithm of our method is shown in Algorithm 1.

Algorithm 1 The C²GSPG Training Algorithm

Require: Initial policy $\pi_\theta = \pi_{\text{ref}}$, dataset \mathcal{D} .

- 1: **for** each training epoch **do**
- 2: **for** each RL step, Sample prompts $\{q\}^M \sim \mathcal{D}$ **do**
- 3: Sample G responses $\{o_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot|q)$.
- 4: For each response, calculate reward r_i and confidence $c_{\theta_{\text{old}},i}$.
- 5: **if** binary reward **then**
- 6: Compute advantage $\tilde{A}_i = \frac{r_i - m}{1 - c_{\theta_{\text{old}},i}}$, where $m = \frac{1}{G} \sum_{i=1}^G r_i$.
- 7: **else**
- 8: Normalize reward: $\hat{r}_i = \sigma(\alpha \cdot r_i) = \frac{1}{1 + \exp(-\alpha \cdot r_i)}$.
- 9: Compute average normalized reward $\hat{m} = \frac{1}{G} \sum_{i=1}^G \hat{r}_i$.
- 10: Compute advantage $\tilde{A}_i = \frac{\hat{r}_i - \hat{m}}{1 - c_{\theta_{\text{old}},i}}$.
- 11: **end if**
- 12: **for** each RL epoch, Sample a mini-batch from rollouts **do**
- 13: Compute confidence $c_{\theta,i}$ with current policy π_θ .
- 14: **if** non binary reward **then**
- 15: Determine $\mathbb{I}_i(\beta) = \begin{cases} \beta, & \text{if } \text{sign}(\hat{r}_i - \hat{m}) = \text{sign}(\hat{r}_i - c_{\theta,i}), \\ 0, & \text{otherwise.} \end{cases}$
- 16: **end if**
- 17: Update policy parameters θ using (6) (binary) or (10) (non binary).
- 18: **end for**
- 19: **end for**
- 20: **end for**
- 21: **return** Final policy parameters θ^* .

The key hyperparameters for our method are detailed in Table 6. To generate settings, we exclude the clipping mechanism in GSPO and fix the F_{norm} parameter to 1 in GPG. For the mathematical reasoning task, the model is trained for 15 epochs (approximately 100 steps). The final reported performance is the average of scores from the 80th, 90th, and 100th steps. Furthermore, to mitigate variance on the smaller AMC 2023 and AIME 2024 test sets, we report the average performance over 8 independent runs for these specific benchmarks. For the logical reasoning task, the model is trained for 5 epochs for all methods. To assess performance, we evaluate the model every 20 steps on separate test sets for each level, and the final reported score is the average of the last three evaluation checkpoints. Furthermore, the low clip ratio and high clip ratio in GRPO and AR-Lopti are specifically set to 0.20 and 0.24 following (Yang et al., 2025b) on this task. Moreover, we apply a KL divergence penalty (with weight $\gamma = 0.001$) to all baselines on the logical reasoning task. For the mathematical reasoning task, we omit it for our method and GPG, but retain it for all other baselines with the same coefficient. The other hyperparameters are mostly set as the default values in VERL.

Greedy search v.s. sampling for sequence generation and evaluation. For the results reported in Tables 3- 5 we use greedy decoding to generate sequences. Accuracy, Expected Calibration Error (ECE), and Brier Score (BS) are computed based on these greedy outputs and their corresponding model confidence scores. For Figures 4 and 1(a), we use a sampling-based decoding strategy with temperature equals 1.0. This allows visualization of model calibration across a broader confidence range, including low-confidence predictions that are less frequent under greedy decoding. So, we can have a more comprehensive view of model calibration behavior.

⁴The VERL framework is available at: <https://github.com/volcengine/verl>

Table 6: Comparison of Key Hyperparameters for Mathematical and Logical Reasoning Tasks

Parameter	Mathematical	Logic
Base Model	Qwen2.5-Math-1.5B	Qwen2.5-3B-Instruct
max_prompt_length	1024	400
max_response_length	3000	4096
total_epochs	15	5
learning_rate	1e-6	1e-6
train_batch_size	1024	64
ppo_mini_batch_size	256	256
ppo_micro_batch_size	16	32
kl_loss_coef	0	0.001
rollout_n	4	8
rollout_temperature	1.0	0.7
beta	0.5	0.03
clip method	no clip	beta clip
n_gpus	8	4
test_freq	5	20

B MORE EXPERIMENT RESULTS

B.1 RESULTS ON MATHEMATICAL REASONING TASK

Figure 6(a) reports the calibration performance on the mathematical training set, which can conclude similar results with Figure 1(a). All methods produce monotonically increasing Fraction of Positives curves (top) with respect to predicted probabilities. However, the curve of our method lies closest to the perfect-calibration diagonal, whereas those of the baselines consistently fall below it, indicating that they tend to be overconfident. Furthermore, the accompanying histogram of predicted probabilities (bottom) shows that our method yields a more dispersed distribution. This suggests a superior ability to differentiate confidence levels, assigning nuanced probabilities rather than collapsing towards extreme values.

B.2 RESULTS ON LOGICAL REASONING TASK

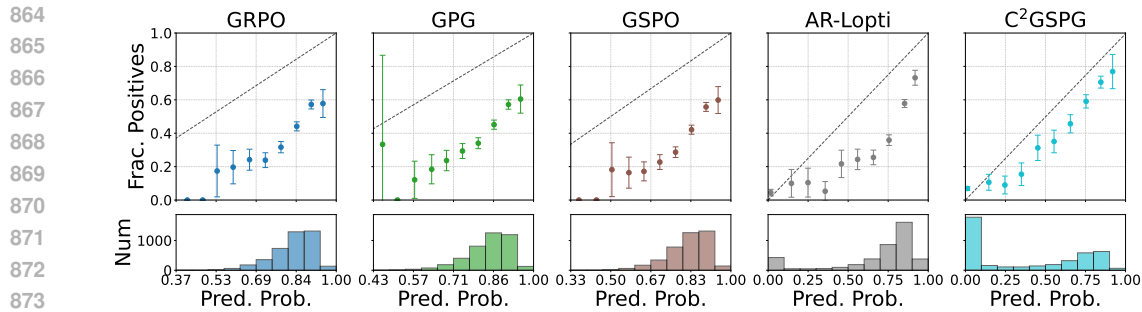
Figure 6(b) presents the calibration dynamics of the models on the K&K training set. In conjunction with Figure 4, we observe that the calibration curves on both the training and test sets are highly consistent, exhibiting a monotonic increasing trend and remaining close to the perfect-calibration diagonal. These results indicate that our method effectively mitigates the issue of model overconfidence while achieving strong generalization performance.

B.3 REWARD FORMULATION FOR LOGICAL REASONING

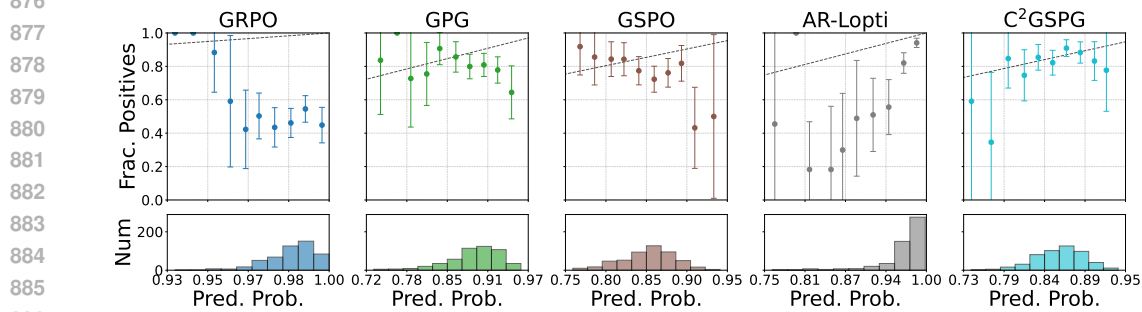
The K&K logic puzzles dataset is based on the classic “Knights and Knaves” scenario. In this setting, each inhabitant of an island is either a Knight (who always tells the truth) or a Knave (who always lies). The model’s objective is to deduce the true identity of each character from their statements.

The complexity of this task is reflected in its unique reward structure, which generates a composite score from the set $\{-3, -1, -0.5, 3\}$. This score is calculated based on two distinct criteria:

- **Formatting Correctness:** The model is required to produce a Chain-of-Thought (CoT) rationale enclosed within `<think>` tags, followed by the final solution in `<answer>` tags. Adherence to this format yields +1 point, while any deviation results in a penalty of -1 point.
- **Answer Accuracy:** The final answer is graded on its correctness. A completely correct solution receives +2 points. A partially correct answer incurs a penalty of -1.5 points, and a fully incorrect answer is penalized by -2 points.



(a) Calibration performance of various methods on mathematical reasoning training set



(b) Calibration performance of various methods on K&K training set

Figure 6: Calibration performance of various methods on the mathematical reasoning and K&K training dataset. Each panel consists of two subplots. The top plot shows the Fraction of Positives against the model’s predicted confidence, with the dashed line representing perfect calibration. The confidence histogram (bottom) illustrates the distribution of predicted confidence levels.

The final reward is the sum of the scores from these two criteria. This multi-faceted reward signal is designed to evaluate both the model’s reasoning process and the accuracy of its conclusions.

B.4 CASE STUDIES IN MATHEMATICAL REASONING

Case Study 1

Question: Alisa biked 12 miles per hour for 4.5 hours. Stanley biked at 10 miles per hour for 2.5 hours. How many miles did Alisa and Stanley bike in total? Let’s think step by step and output the final answer within `[answer]`. (Ground-truth answer is 79)

GRPO Answer (Confidence: 0.9428): To find the total distance biked by Alisa and Stanley, we can follow these steps:

1. Calculate the distance biked by Alisa.
2. Calculate the distance biked by Stanley.
3. Add the two distances together to get the total distance.

The formula to calculate distance is:

$$\text{Distance} = \text{Speed} \times \text{Time}$$

Let’s break it down:

1. Alisa’s distance:

$$\text{Distance}_{\text{Alisa}} = 12 \text{ miles/hour} \times 4.5 \text{ hours}$$

918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971

2. Stanley's distance:

$$\text{Distance}_{\text{Stanley}} = 10 \text{ miles/hour} \times 2.5 \text{ hours}$$

3. Total distance:

$$\text{Total Distance} = \text{Distance}_{\text{Alisa}} + \text{Distance}_{\text{Stanley}}$$

The total distance biked by Alisa and Stanley is 83 miles. The answer is: **83**

GPG Answer (Confidence: 0.9341): To find the total distance biked by Alisa and Stanley, we can follow these steps:

1. Calculate the distance biked by Alisa.
2. Calculate the distance biked by Stanley.
3. Add the two distances together to get the total distance.

The formula to calculate distance is:

$$\text{Distance} = \text{Speed} \times \text{Time}$$

Let's break it down:

1. Alisa's distance:

$$\text{Distance}_{\text{Alisa}} = 12 \text{ miles/hour} \times 4.5 \text{ hours}$$

2. Stanley's distance:

$$\text{Distance}_{\text{Stanley}} = 10 \text{ miles/hour} \times 2.5 \text{ hours}$$

3. Total distance:

$$\text{Total Distance} = \text{Distance}_{\text{Alisa}} + \text{Distance}_{\text{Stanley}}$$

The total distance biked by Alisa and Stanley is 89 miles. The answer is: **89**

GSPO Answer (Confidence: 0.9162): To find the total distance biked by Alisa and Stanley, we can follow these steps:

1. Calculate the distance biked by Alisa.
2. Calculate the distance biked by Stanley.
3. Add the two distances together to get the total distance.

The formula to calculate distance is:

$$\text{Distance} = \text{Speed} \times \text{Time}$$

Let's break it down:

1. Alisa's distance:

$$\text{Distance}_{\text{Alisa}} = 12 \text{ miles/hour} \times 4.5 \text{ hours}$$

2. Stanley's distance:

$$\text{Distance}_{\text{Stanley}} = 10 \text{ miles/hour} \times 2.5 \text{ hours}$$

3. Total distance:

$$\text{Total Distance} = \text{Distance}_{\text{Alisa}} + \text{Distance}_{\text{Stanley}}$$

The total distance biked by Alisa and Stanley is 83 miles. The answer is: **83**

AR-Lopti Answer (Confidence: 0.9074): To find the total distance biked by Alisa and Stanley, we can follow these steps:

972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025

1. Calculate the distance biked by Alisa.
2. Calculate the distance biked by Stanley.
3. Add the two distances together to get the total distance.

The formula to calculate distance is:

$$\text{Distance} = \text{Speed} \times \text{Time}$$

Let's break it down:

1. Alisa's distance:

$$\text{Distance}_{\text{Alisa}} = 12 \text{ miles/hour} \times 4.5 \text{ hours}$$

2. Stanley's distance:

$$\text{Distance}_{\text{Stanley}} = 10 \text{ miles/hour} \times 2.5 \text{ hours}$$

3. Total distance:

$$\text{Total Distance} = \text{Distance}_{\text{Alisa}} + \text{Distance}_{\text{Stanley}}$$

Now let's calculate these values using Python. The total distance biked by Alisa and Stanley is 83 miles. The answer is: **83**

C²GSPG Answer (Confidence: 0.9086): First, we calculate the distance Alisa biked:

$$\text{Distance}_{\text{Alisa}} = \text{Speed}_{\text{Alisa}} \times \text{Time}_{\text{Alisa}} = 12 \text{ miles/hour} \times 4.5 \text{ hours}$$

Next, we calculate the distance Stanley biked:

$$\text{Distance}_{\text{Stanley}} = \text{Speed}_{\text{Stanley}} \times \text{Time}_{\text{Stanley}} = 10 \text{ miles/hour} \times 2.5 \text{ hours}$$

Then we add these distances together to find the total distance biked:

$$\text{Total Distance} = \text{Distance}_{\text{Alisa}} + \text{Distance}_{\text{Stanley}}$$

Now we perform the calculations:

$$\text{Distance}_{\text{Alisa}} = 12 \times 4.5 = 54 \text{ miles}$$

$$\text{Distance}_{\text{Stanley}} = 10 \times 2.5 = 25 \text{ miles}$$

$$\text{Total Distance} = 54 + 25 = 79 \text{ miles}$$

So the total distance biked by Alisa and Stanley is 79 miles. The answer is: **79**

B.5 CASE STUDIES IN LOGICAL REASONING

Case Study 1

Question: A very special island is inhabited only by knights and knaves. Knights always tell the truth, and knaves always lie. You meet 3 inhabitants: Emily, Aria, and Abigail. As Emily put it, "Emily is a knight or Aria is a knave". Aria said that "If Aria is a knight then Abigail is a knight". Abigail stated, "Aria is a knight if and only if Emily is a knave". So who is a knight and who is a knave? (Ground-truth answer is **Emily is a knave. Aria is a knight. Abigail is a knight.**)

GRPO Answer (Confidence: 0.9947): <think> Emily's statement is "Emily is a knight or Aria is a knave". If Emily were a knave, her statement would be false, but since she is a knight, her statement is true. This means that Emily is a knight. Since Emily is a knight, her statement "Emily is a knight or Aria is a knave" is true, so Aria must be a knight. Since Aria

1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079

is a knight, her statement “If Aria is a knight then Abigail is a knight” is true, so Abigail must be a knight. Finally, Abigail’s statement is “Aria is a knight if and only if Emily is a knave”. Since Aria is a knight and Emily is a knight, this statement is false, so Abigail is a knave. </think>

<answer> **Emily is a knight. Aria is a knight. Abigail is a knave.** </answer>

GPG Answer (Confidence: 0.9309): <think>Let’s analyze the statements one by one and see if we can determine the nature of each inhabitant.

1. **Abigail’s statement:** “Aria is a knight if and only if Emily is a knave.”
 - If Abigail is a knight, then Aria must be a knave and Emily must be a knight. But this contradicts the statement that Aria is a knight if and only if Emily is a knave. So, Abigail must be a knave.
 - If Abigail is a knave, then Aria must not be a knight if Emily is a knave, and Aria must be a knight if Emily is a knight.
2. **Aria’s statement:** “If Aria is a knight then Abigail is a knight.”
 - Since we know Abigail is a knave, and Aria’s statement is that if Aria is a knight, then Abigail is a knight, Aria must be a knave.
3. **Emily’s statement:** “Emily is a knight or Aria is a knave.”
 - Since Aria is a knave, the statement “Emily is a knight or Aria is a knave” is true. Therefore, Emily must be a knight.
4. **Aria’s statement:** “If Aria is a knight then Abigail is a knight.”
 - Since Aria is a knave, the statement “If Aria is a knight then Abigail is a knight” is vacuously true. So, Aria is a knave.
5. **Abigail’s statement:** “Aria is a knight if and only if Emily is a knave.”
 - Since Aria is a knave and Emily is a knight, Abigail’s statement is true. Therefore, Abigail is a knight.
6. **Aria’s statement:** “If Aria is a knight then Abigail is a knight.”
 - Since Aria is a knave, the statement “If Aria is a knight then Abigail is a knight” is vacuously true. So, Aria is a knave.
7. **Emily’s statement:** “Emily is a knight or Aria is a knave.”
 - Since Emily is a knight and Aria is a knave, the statement is true. Therefore, Emily is a knight.

</think>

<answer> **Emily is a knight, Aria is a knave, Abigail is a knight.** </answer>

GSPO Answer (Confidence: 0.8906): <think>First, let’s analyze Abigail’s statement: “Aria is a knight if and only if Emily is a knave.” This is a biconditional statement, so it must be either both true or both false. If Abigail were a knight, then this statement would be true, meaning Aria must be a knight and Emily must be a knave. However, if Emily were a knave, then the statement “Emily is a knight or Aria is a knave” would be true, which would mean Emily is a knight, creating a contradiction. Therefore, Abigail must be a knave.

Now, let’s move to Aria’s statement: “If Aria is a knight, then Abigail is a knight.” Since Abigail is a knave, this conditional statement must be a true statement, meaning Aria must be a knight.

Next, let’s examine Emily’s statement: “Emily is a knight or Aria is a knave.” Since we already know Aria is a knight, this statement must be true, so Emily must be a knight.

Finally, let’s verify the consistency of these conclusions:

1. Emily is a knight (as verified by her statement).
2. Aria is a knight (as verified by her statement).
3. Abigail is a knave (as verified by her statement).

1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133

All statements are consistent with each other and with the rules of the island.

</think>

<answer> **Emily is a knight. Aria is a knight. Abigail is a knave.** </answer>

AR-Lopti Answer (Confidence: 0.9933): <think>Let’s analyze the statements step by step:

1. Emily says, “Emily is a knight or Aria is a knave.”
2. Aria says, “If Aria is a knight then Abigail is a knight.”
3. Abigail says, “Aria is a knight if and only if Emily is a knave.”

First, let’s consider Emily’s statement: “Emily is a knight or Aria is a knave.” This is a tautology and must be true. Therefore, Emily must be a knight.

Now, let’s look at Aria’s statement: “If Aria is a knight then Abigail is a knight.” This is a valid statement because if Aria is a knight, then Abigail must be a knight. Therefore, Aria must be a knight.

Finally, let’s consider Abigail’s statement: “Aria is a knight if and only if Emily is a knave.” Since Emily is a knight, the condition “if and only if Emily is a knave” is false. Therefore, Abigail must be a knave.

So, we can conclude:

- Emily is a knight.
- Aria is a knight.
- Abigail is a knave.

Therefore, Emily is a knight, Aria is a knight, and Abigail is a knave.

</think>

<answer> **Emily is a knight. Aria is a knight. Abigail is a knave.** </answer>

C²GSPG Answer (Confidence: 0.9065): <think>Let’s analyze the statements step by step. Let’s analyze the statements step by step: 1. Emily says, “Emily is a knight or Aria is a knave.” 2. Aria says, “If Aria is a knight then Abigail is a knight.” 3. Abigail says, “Aria is a knight if and only if Emily is a knave.”

First, let’s consider Aria’s statement. Aria’s statement is a conditional statement: “If Aria is a knight, then Abigail is a knight.” This is a true statement because any conditional statement “If P, then Q” is true unless P is true and Q is false. Since Aria is making a conditional statement, and Aria is a knight, the antecedent (Aria is a knight) is true, and the consequent (Abigail is a knight) must also be true for the whole statement to be true. Therefore, Abigail must be a knight.

Now let’s consider Abigail’s statement: “Aria is a knight if and only if Emily is a knave.” Since we already determined that Abigail is a knight, the statement “Aria is a knight if and only if Emily is a knave” must be true. This means that if Aria is a knight, then Emily must be a knave, and if Emily is a knave, then Aria must be a knight.

Now let’s consider Emily’s statement: “Emily is a knight or Aria is a knave.” Since Emily is a knave and Aria is a knight, the disjunction “Emily is a knight or Aria is a knave” is false. Therefore, Emily’s statement is false, which means Emily is a knave.

In conclusion:

- Emily is a knave.
- Aria is a knight.
- Abigail is a knight.

</think>

<answer> **Emily is a knave. Aria is a knight. Abigail is a knight.** </answer>

C USE OF LARGE LANGUAGE MODELS (LLMs)

In this work, LLMs are used for language refinement of the paper and for assisting in verifying the correctness of code implementations.

D ADDED EXPERIMENT RESULTS

D.1 SCALABILITY AND ROBUSTNESS ON LARGER MODELS

To further evaluate the scalability and robustness of our method, we conduct experiments on larger-scale models. Following Shi et al. (2025) and Yang et al. (2025b), we fine-tune the Qwen2.5-7B model⁵ for the mathematical reasoning task, and the Qwen2.5-7B-Instruct-1M model⁶ for the logical reasoning task, respectively. The batch size is halved due to computational constraints. For the mathematical reasoning task, training is slightly extended to 150 steps to better demonstrate robustness. Other settings remained unchanged. Results in Table 7 and Table 8 show that our method outperforms baselines consistently on larger LLMs, demonstrating its scalability.

Table 7: Comparison of various methods applied to 7B models on mathematical reasoning benchmarks. The row of “Base” shows the performance of the model without any post-training. In each column, the best result is highlighted in **bold**.

Method	Accuracy (%) by Dataset						Overall Performance		
	GSM8K	MATH500	AMC2023	AIME2024	Olympiad	Minerva	Acc (↑)	BS (↓)	ECE (↓)
Base	61.26	46.20	21.25	6.25	16.30	9.93	26.86	0.598	0.632
GRPO	89.60	72.04	43.87	16.33	25.10	23.82	45.13	0.531	0.535
GPG	89.63	73.24	51.56	13.00	24.62	23.75	45.97	0.515	0.520
GSPO	88.75	72.44	53.19	14.67	24.56	22.35	45.99	0.520	0.524
AR-Lopti	88.72	73.08	47.31	14.42	24.65	24.26	45.41	0.522	0.528
C ² GSPG	90.60	71.56	52.12	20.67	24.44	22.94	47.06	0.511	0.515

Table 8: Performance comparison of various methods on the K&K dataset for 7B models. The row of “Base” shows the performance of the model without post-training. In each column, the best result is highlighted in **bold**.

Method	Accuracy (%) by # Characters					Overall Performance		
	3	4	5	6	7	Acc (↑)	BS (↓)	ECE (↓)
Base	23.00	17.00	7.00	10.00	3.00	12.00	0.460	0.610
GRPO	79.00	72.40	56.40	56.60	42.40	61.36	0.254	0.308
GPG	90.60	88.20	73.20	69.60	53.60	75.04	0.190	0.194
GSPO	84.00	88.00	75.20	72.00	67.20	77.28	0.157	0.171
AR-Lopti	82.60	87.60	71.60	68.00	58.40	73.64	0.180	0.214
C ² GSPG	95.60	92.80	85.80	77.00	73.00	84.84	0.126	0.124

D.2 ABLATION STUDY

The Effect of Different Components in Our Method. Our C²GSPG method includes a GSPG framework and a BCE regularizer. To individually validate the effectiveness of these two modules, we design an ablation study on the K&K dataset using the Qwen2.5-3B-Instruct model. In particular, for each RL method (including baselines and our GSPG framework), we augment it by incorporating our BCE regularizer. The results in Table 9 show that a) the BCE regularizer is universally effective, which improves various methods consistently, and b) the GSPG framework outperforms the baselines consistently.

The Effect of Hyperparameter β . We fine-tune Qwen2.5-3B-Instruct on the K&K dataset with $\beta \in \{0.01, 0.03, 0.05\}$ to study its impact. The results in Table 10 show that an improperly set β can lead to under- or over-regularization, and our setting ($\beta = 0.03$) leads to better performance. Notably, the models using sub-optimal settings ($\beta = 0.01$ and 0.05) still outperform the baselines, demonstrating the robustness of our method.

⁵The model is available at: <https://huggingface.co/Qwen/Qwen2.5-7B>

⁶The model is available at: <https://huggingface.co/Qwen/Qwen2.5-7B-Instruct-1M>

Table 9: The comparison for various methods on the K&K dataset. The row of “Base” shows the performance of Qwen2.5-3B-Instruct without post-training. In each column, the best result is **bold**.

Method	Accuracy (%) by # Characters					Overall Performance		
	3	4	5	6	7	Acc (↑)	BS (↓)	ECE (↓)
Base	12.00	13.00	2.00	2.00	0.00	5.80	0.554	0.705
GRPO	58.67	44.93	34.53	31.87	20.53	38.11	0.405	0.497
GRPO+BCE	66.60	57.07	45.80	41.13	29.73	48.07	0.335	0.409
GPG	69.53	64.80	49.47	43.20	35.93	52.59	0.363	0.363
GPG+BCE	70.93	66.07	53.07	43.87	36.33	54.05	0.337	0.330
GSP0	71.13	66.40	55.40	48.20	40.07	56.24	0.291	0.302
GSP0+BCE	77.07	76.53	61.80	55.73	47.27	63.68	0.231	0.233
AR-Lopti	73.20	64.93	54.13	49.47	41.33	56.61	0.282	0.343
AR-Lopti+BCE	78.40	81.80	70.27	66.13	58.93	71.11	0.192	0.225
GSPG	79.50	75.50	67.00	57.00	50.00	65.80	0.215	0.256
GSPG+BCE(Our C ² GSPG)	86.00	88.00	78.60	73.27	69.00	78.97	0.136	0.126

Table 10: Ablation study on different values of β on the K&K logic puzzles dataset.

Beta (β)	Accuracy (%) by # Characters					Overall Performance		
	3	4	5	6	7	Acc (↑)	BS (↓)	ECE (↓)
0.01	82.00	83.00	70.00	64.33	61.00	72.07	0.169	0.171
0.03	86.00	88.00	78.60	73.27	69.00	78.97	0.136	0.126
0.05	81.00	83.00	73.00	65.33	58.33	72.13	0.168	0.132

The Effect of Hyperparameter α . We study the impact of the reward transformation hyperparameter, $\alpha \in \{1, 3, 5\}$, with results in Table 11. When α is too large, the reward transformation converges to a step function, reducing the distinguishability across different negative rewards. When α is too small, the negative reward after transformation is far from zero, leading to a higher probability of gradient conflict. The results show that the setting of $\alpha = 3$ is reasonable, and setting $\alpha \geq 3$ leads to relatively stable performance.

Table 11: Ablation study on different values of alpha on the K&K logic puzzles dataset finetuned by Qwen2.5-3B-Instruct.

α	Reward Mapping	Accuracy (%) by # Characters					Overall Performance		
		3	4	5	6	7	Acc (↑)	BS (↓)	ECE (↓)
1	{0, 0.268941, 0.377541, 1}	77.33	83.67	71.67	67.00	60.00	71.93	0.181	0.209
3	{0, 0.047426, 0.182426, 1}	86.00	88.00	78.60	73.27	69.00	78.97	0.136	0.126
5	{0, 0.006693, 0.075858, 1}	84.33	87.33	80.00	75.33	66.33	78.67	0.142	0.145

D.3 THE IMPACT OF IMPROVED CONFIDENCE ON INFERENCE-TIME PERFORMANCE

To investigate whether our method’s improved sequence confidence enhances test-time scaling, we conduct a Best-of-N (BoN) experiment on the K&K dataset. In this setup, we use the model’s own sequence confidence as the ranker to select the best response from a set of candidates. Specifically, for each method, we sample $N \in \{1, 4, 8, 16\}$ responses. Then, we compare the average accuracy across all samples (Avg@N) with that achieved by selecting the most confident sample (Best@N). The results in Table 12 show that our method works well when using the BoN strategy, while the baselines without confidence calibration fail to achieve consistent improvements. This phenomenon indicates that our confidence calibration method helps improve inference-time self-awareness.

Table 12: Performance comparison (%) using sampling. The Gap (Δ) column is calculated as ‘Best-Avg’ to quantify the difference.

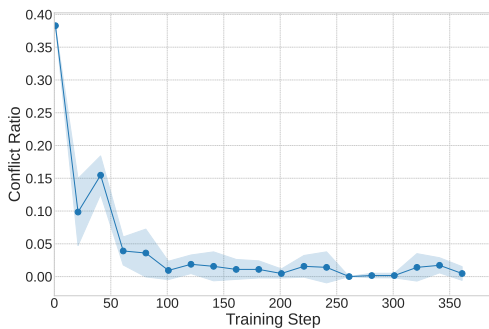
Method	@1	@4			@8			@16		
		Avg	Best	Δ	Avg	Best	Δ	Avg	Best	Δ
GRPO	38.20	37.05	36.60	-0.45	37.05	37.20	+0.15	37.05	37.20	+0.15
GPG	40.20	40.30	39.60	-0.70	41.43	41.80	+0.37	42.03	43.00	+0.97
GSPO	54.60	55.90	56.20	+0.30	56.27	56.40	+0.13	56.01	55.80	-0.21
AR-Lopti	56.40	56.30	56.80	+0.50	57.00	57.20	+0.20	57.13	58.40	+1.27
C ² GSPG	77.80	79.05	84.60	+5.55	80.27	85.40	+5.13	80.20	86.40	+6.20

D.4 THE EFFECT OF OUR ADAPTIVE CLIPPING STRATEGY

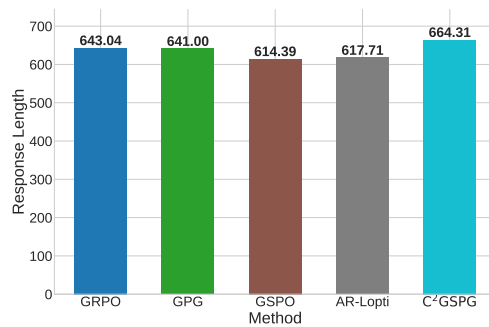
Our adaptive clipping strategy is motivated by an analysis of the policy gradient (Equation (9)), which revealed potential conflicts between reward scores and model confidence that can cause training instability. As shown in Figure 7(a), we plot the proportion of samples requiring clipping (the *conflict ratio*) over time. The ratio is high initially but rapidly decays towards zero as training progresses. This demonstrates that the strategy is critical for ensuring stability in the early stages of training.

D.5 THE REASONING LENGTH OF DIFFERENT METHODS

We also analyze the response lengths on the MATH dataset. As shown in Figure 7(b) and an example in Appendix B.4, our method generates slightly longer reasoning chains on average compared to the baselines. In our opinion, this phenomenon is due to confidence calibration – a more self-aware model may adopt a more deliberate, step-by-step process to construct a reliable solution.



(a) Conflict Ratio of C²GSPG VS. Step



(b) Response Length of Different Methods

Figure 7: (a) depicts the evolution of the percentage of training samples on the K&K dataset where the gradients of the policy term and the regularizer term conflict. (b) provides a comparative analysis of the response lengths generated by models trained with different methods on the MATH dataset.

D.6 COMPARISON OF REINFORCEMENT FINE-TUNING AGAINST SUPERVISED FINE-TUNING

We also train a baseline on the K&K dataset using Supervised Fine-Tuning (SFT). This SFT model is trained by directly optimizing the log-likelihood that the output is correct for each problem. Consistent with the RL-based method, we set the batch size to 256 and train for 5 epochs. The results in Table 13 show that the SFT baseline performs worse than RL-based methods across all metrics of accuracy and calibration. This demonstrates that merely optimizing the log-likelihood of correct answers is insufficient for achieving competitive performance on reasoning tasks.

Table 13: The comparison for RFT-based and SFT methods on the K&K dataset. The row of “Base” shows the performance of Qwen2.5-3B-Instruct without post-training.

Method	Accuracy (%) by # Characters					Overall Performance		
	3	4	5	6	7	Acc (↑)	BS (↓)	ECE (↓)
Base	12.00	13.00	2.00	2.00	0.00	5.80	0.554	0.705
SFT	38.00	33.00	18.00	18.00	8.00	23.00	0.506	0.623
GRPO	58.67	44.93	34.53	31.87	20.53	38.11	0.405	0.497

D.7 MORE CASES ON DIFFERENT TASKS

In Appendix B.4 and B.5, we have presented failure cases for the baseline models. These models exhibit high confidence when generating incorrect outputs, which illustrates the issue of overconfidence. We provide failure cases of our method in this section. When generating incorrect outputs, our model is consistently less confident than baselines. The baselines lack self-awareness – their confidence and correctness are inconsistent in general.

Case Study 1

Question: Abraham owns 80 square meters of unused land. He sold half of the land for \$50, and after a month, he sold another $\frac{1}{4}$ of his land for \$30. He then sold the remaining land for \$3 per square meter. How much money will he be able to earn after selling all his unused land? Let’s think step by step and output the final answer within answer. (Ground-truth answer is 170)

GRPO Answer (Confidence: 0.8317): First, let’s calculate how much land Abraham sold in each transaction:

1. He sold half of his land for \$50. Since his total land is 80 square meters, the area of land he sold in the first transaction is $\frac{80}{2} = 40$ square meters.
2. He sold another $\frac{1}{4}$ of his land for \$30. So, the area of land he sold in the second transaction is $\frac{80}{4} = 20$ square meters.
3. The remaining land is $80 - 40 - 20 = 20$ square meters. He sold the remaining land for \$3 per square meter, so he earned $20 \times 3 = 60$ dollars.

Adding up the earnings from all the transactions, we get:

$$50 + 30 + 60 = 140$$

The total amount of money Abraham will be able to earn after selling all his unused land is 140. The answer is: **140 (Wrong answer)**

GPG Answer (Confidence: 0.8457): First, let’s calculate how much land Abraham sold in each transaction:

1. He sold half of his land for \$50. Since his total land is 80 square meters, the area of land he sold in the first transaction is $\frac{80}{2} = 40$ square meters.
2. He sold another $\frac{1}{4}$ of his land for \$30. So, the area of land he sold in the second transaction is $\frac{80}{4} = 20$ square meters.
3. The remaining land is $80 - 40 - 20 = 20$ square meters, which he sold for \$3 per square meter.

Now, let’s calculate the amount of money he earned from each transaction:

1. From the first transaction: $40 \text{ meters}^2 \times \frac{50}{40} = 40 \text{ meters}^2 \times \frac{50}{40} = 50$ dollars.
2. From the second transaction: $20 \text{ meters}^2 \times \frac{30}{20} = 20 \text{ meters}^2 \times \frac{30}{20} = 30$ dollars.
3. From the third transaction: $20 \text{ meters}^2 \times 3 = 60$ dollars.

1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403

So, the total amount of money he will earn after selling all his unused land is $50 + 30 + 60 = 140$ dollars. The final answer is: $\boxed{140}$. The answer is: **140 (Wrong answer)**

GSPO Answer (Confidence: 0.8561): To find the final price of Stephen's groceries, we need to consider the following:

1. The initial cost of the groceries: \$40.00
2. The 25% delivery fee on the initial cost
3. The delivery fee of \$3.00
4. The tip of \$4.00

First, we calculate the delivery fee:

$$\text{Delivery Fee} = 0.25 \times 40.00 = 10.00$$

Next, we add the delivery fee and the tip to the initial cost and the delivery fee:

$$\text{Final Price} = 40.00 + 10.00 + 3.00 + 4.00$$

Now we can calculate the final price:

$$\text{Final Price} = 40.00 + 10.00 + 3.00 + 4.00 = 57.00$$

The final price of Stephen's groceries is $\boxed{57.00}$.

The answer is: **57 (Wrong answer)**

AR-Lopti Answer (Confidence: 0.8456): First, let's calculate how much land Abraham sold in each transaction:

1. He sold half of his land for \$50. Since his total land is 80 square meters, the area of land he sold in the first transaction is $\frac{80}{2} = 40$ square meters.
2. He sold another $\frac{1}{4}$ of his land for \$30. So, the area of land he sold in the second transaction is $\frac{80}{4} = 20$ square meters.
3. The remaining land is $80 - 40 - 20 = 20$ square meters, which he sold for \$3 per square meter.

Now, let's calculate the amount of money he earned from each transaction:

1. From the first transaction: $40 \text{ meters}^2 \times \frac{50}{40} = 40 \text{ meters}^2 \times \frac{50}{40} = 50$ dollars.
2. From the second transaction: $20 \text{ meters}^2 \times \frac{30}{20} = 20 \text{ meters}^2 \times \frac{30}{20} = 30$ dollars.
3. From the third transaction: $20 \text{ meters}^2 \times 3 = 60$ dollars.

So, the total amount of money he will earn after selling all his unused land is $50 + 30 + 60 = 140$ dollars. The final answer is: $\boxed{140}$ The answer is: **140 (Wrong answer)**

C²GSPG Answer (Confidence: 0.5106): First, Abraham owns a total of 80 square meters of land.

1. He sold half of the land, which is $\frac{80}{2} = 40$ square meters, for \$50. Therefore, the price per square meter for this part of the land is $50/40 = 1.25$ per square meter. However, we notice that this part of the land was sold for \$50 in total.
2. He sold another $\frac{1}{4}$ of the land, which is $\frac{80}{4} = 20$ square meters, for \$30. Therefore, the price per square meter for this part of the land is $30/20 = 1.50$ per square meter. However, we notice that this was sold for \$30 in total.
3. The remaining land he sold for \$3 per square meter. The remaining land is $80 - 40 - 20 = 20$ square meters. So, he sold these 20 square meters for $20 \times 3 = 60$ dollars.

1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457

Now we can calculate the total money he earned from selling the land: $\$50$ for the first part + $\$30$ for the second part + $\$60$ for the remaining part = $\$50 + \$30 + \$60 = \140 .

However, we notice that the price per square meter mentioned in the first two steps ($\$1.25$ and $\$1.50$) doesn't match the direct price given ($\$50$ for 40 square meters or $\$30$ for 20 square meters). This means the problem statement might have inconsistent information initially, but it can be solved by carefully considering the given direct prices and their implications.

Since the problem implies consistent land sales but the per-meter calculations don't match, let's validate the total earnings directly: $\$50 + \$30 + \$60 = \140 remains the correct manual calculation. Atlanta's answer suggests earning $\$140$.

The final answer is: The answer is: **140 (Wrong answer)**

Case Study 2

Question: Ricardo grows tomatoes and eggplants in his garden. Each tomato plant yields 22 tomatoes while each plant of eggplant yields 4 eggplants. He planted 5 tomato plants and 8 plants of eggplant. How many fruits can Ricardo get from his plants? Let's think step by step and output the final answer within . (Ground-truth answer is 142)

GRPO Answer (Confidence: 0.9141): To determine the total number of fruits Ricardo can get from his plants, we need to calculate the number of tomatoes and eggplants separately and then sum them up.

1. Each tomato plant yields 22 tomatoes, and he has 5 tomato plants.
2. Each eggplant plant yields 4 eggplants, and he has 8 eggplant plants.

So, the total number of tomatoes is 22×5 and the total number of eggplants is 4×8 . Adding these two quantities together will give us the total number of fruits.

The total number of fruits Ricardo can get from his plants is . The answer is: **146 (Wrong answer)**

GPG Answer (Confidence: 0.9257): To determine the total number of fruits Ricardo can get from his plants, we need to calculate the number of tomatoes and eggplants separately and then sum them up.

1. Each tomato plant yields 22 tomatoes, and he has 5 tomato plants.
2. Each eggplant plant yields 4 eggplants, and he has 8 eggplant plants.

So, the total number of tomatoes is 22×5 and the total number of eggplants is 4×8 . Adding these two quantities will give the total number of fruits.

The total number of fruits Ricardo can get from his plants is . The answer is: **154 (Wrong answer)**

GSPO Answer (Confidence: 0.9201): To determine the total number of fruits Ricardo can get from his plants, we need to calculate the number of tomatoes and eggplants separately and then sum them up.

1. Each tomato plant yields 22 tomatoes, and he has 5 tomato plants.
2. Each eggplant plant yields 4 eggplants, and he has 8 eggplant plants.

So, the total number of tomatoes is 22×5 and the total number of eggplants is 4×8 . The total number of fruits is the sum of these two quantities.

Ricardo can get a total of fruits from his plants. The answer is: **154 (Wrong answer)**

AR-Lopti Answer (Confidence: 0.8337): To determine the total number of fruits Ricardo can get from his plants, we need to calculate the number of tomatoes and eggplants separately and then sum them up.

1. Each tomato plant yields 22 tomatoes, and he has 5 tomato plants.

1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511

2. Each eggplant plant yields 4 eggplants, and he has 8 eggplant plants.

So, the total number of tomatoes is 22×5 and the total number of eggplants is 4×8 . The total number of fruits is the sum of the tomatoes and eggplants.

The total number of tomatoes is $22 \times 5 = 110$.

The total number of eggplants is $4 \times 8 = 32$.

So, the total number of fruits is $110 + 32 = 142$.

The total number of fruits Ricardo can get from his plants is 142. The answer is: **142 (Correct answer)**

C²GSPG Answer (Confidence: 0.5934): First, we need to calculate the total number of tomatoes produced by the 5 tomato plants. Since each tomato plant yields 22 tomatoes, the total number of tomatoes is 5×22 .

Then, we need to calculate the total number of eggplants produced by the 8 eggplant plants. Since each eggplant plant yields 4 eggplants, the total number of eggplants is 8×4 .

Finally, we sum up the total number of tomatoes and eggplants to get the total number of fruits. The total number of fruits is the sum of all the tomatoes and all the eggplants. We will then output the final answer within \square .

Therefore, the total number of fruits Ricardo can get from his plants is 150. The answer is: **150 (Wrong answer)**

Case Study 3

Question: A very special island is inhabited only by knights and knaves. Knights always tell the truth, and knaves always lie. You meet 4 inhabitants: Alexander, Grace, Avery, and Aurora. “Grace is not a knight” - Alexander. Grace expressed that Aurora is a knight and Grace is a knight. Avery was heard saying, “Grace is a knight if and only if Avery is a knight”. “Avery is not a knave,” Aurora claimed. So who is a knight and who is a knave? (Ground-truth answer is Alexander is a knave, Grace is a knight, Avery is a knight, Aurora is a knight)

GRPO Answer (Confidence: 0.9889): `<think>` Alexander’s statement is “Grace is not a knight”, so Alexander must be a knave. Grace’s statement is “Aurora is a knight and Grace is a knight”, which is a contradiction, so Grace must be a knave. Avery’s statement is “Grace is a knight if and only if Avery is a knight”, which is false, so Avery must be a knave. Aurora’s statement is “Avery is not a knave”, which is true, so Aurora must be a knight. `</think>`
`<answer>` **Alexander is a knave, Grace is a knave, Avery is a knave, Aurora is a knight.** `</answer>` **(Wrong answer)**

GPG Answer (Confidence: 0.7802): `<think>` Let’s analyze the statements logically to determine who is a knight and who is a knave. The most effective method is to test a hypothesis and check for contradictions.

1. **Determine Grace’s identity.** Let’s assume Grace is a Knave.

- If Grace is a Knave, then her statement, “Aurora is a knight and Grace is a knight,” must be false. This is consistent.
- Alexander’s statement, “Grace is not a knight,” would be true, which would make Alexander a Knight.
- Now consider Avery’s statement: “Grace is a knight if and only if Avery is a knight.” Since we assume Grace is a Knave, the first part is false. The statement becomes *False* \Leftrightarrow *Avery is a Knight*.
- If Avery were a Knight, his statement must be true, but *False* \Leftrightarrow *True* is False. This is a contradiction.
- If Avery were a Knave, his statement must be false, but *False* \Leftrightarrow *False* is True. This is also a contradiction.
- Since assuming Grace is a Knave leads to an inescapable contradiction, our assumption must be false. Therefore, **Grace must be a Knight.**

1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565

2. Determine the others' identities based on Grace being a Knight.

- **Alexander:** His statement "Grace is not a knight" is now proven to be false. Therefore, **Alexander is a Knave.**
- **Aurora:** Since Grace is a Knight, her statement "Aurora is a knight and Grace is a knight" must be true. This means **Aurora is a Knight.**
- **Avery:** Since Aurora is a Knight, her statement "Avery is not a knave" must be true. This means **Avery is a Knight.**

</think>

<answer> **Alexander is a knave, Grace is a knight, Avery is a knight, Aurora is a knight.**(Correct answer) </answer>

GSPO Answer (Confidence: 0.7292): <think> Let's analyze the statements in a logical sequence to determine each person's identity. The most robust method is to establish one person's identity and then deduce the others.

1. Determine Grace's Identity.

Grace's statement is, "Aurora is a knight and Grace is a knight." This is a compound statement. Let's test the hypothesis that Grace is a knave.

- If Grace were a knave, her statement would be false.
- Avery's statement is, "Grace is a knight if and only if Avery is a knight." If Grace is a knave, this becomes "*False if and only if Avery is a knight.*"
- For this biconditional to be true, Avery must be a knave. But if Avery is a knave, his statement must be false. "*False if and only if False*" is a true statement, which contradicts Avery being a knave.
- For the biconditional to be false, Avery must be a knight. But if Avery is a knight, his statement must be true. "*False if and only if True*" is a false statement, which is consistent with Avery being a knight. So, if Grace is a knave, Avery must be a knight.
- Let's check Aurora's statement with this finding: "Avery is not a knave." Since Avery is a knight, this statement is true, making Aurora a knight.
- Now we check Alexander's statement: "Grace is not a knight." Since we assumed Grace is a knave, this statement is true, making Alexander a knight.
- So far: Grace (Knave), Avery (Knight), Aurora (Knight), Alexander (Knight). Let's re-check Grace's statement. "Aurora is a knight and Grace is a knight." This becomes "*True and False,*" which is False. This is consistent with Grace being a knave.

Let's re-evaluate. My previous reasoning was flawed. A more direct path:

If we assume **Grace is a Knave**, then her statement "Aurora is a knight and Grace is a knight" must be false. Alexander's statement "Grace is not a knight" would be true, making **Alexander a Knight**. Avery's statement becomes "False \Leftrightarrow Avery is a knight". If Avery is a Knight, his statement must be true, but False \Leftrightarrow True is False (a contradiction). If Avery is a Knave, his statement must be false, but False \Leftrightarrow False is True (also a contradiction).

Since assuming Grace is a knave leads to a contradiction, the assumption must be false. Therefore, **Grace must be a Knight.**

2. Deduce the Others.

Now that we know Grace is a Knight, her statements are true.

- From Grace's statement, "Aurora is a knight and Grace is a knight," we can conclude that **Aurora is a Knight.**
- Alexander's statement, "Grace is not a knight," is false. Therefore, **Alexander is a Knave.**
- Since Aurora is a Knight, her statement, "Avery is not a knave," must be true. This means Avery is not a knave, so **Avery is a Knight.**
- We can verify Avery's statement: "Grace is a knight if and only if Avery is a knight." This becomes "*True if and only if True,*" which is a true statement, consistent with Avery being a Knight.

1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619

</think>

<answer> **Alexander is a knave. Grace is a knight. Avery is a knight. Aurora is a knight. (Correct answer)** </answer>

AR-Lopti Answer (Confidence: 0.9960): <think> Alexander says that Grace is not a knight. If Alexander is a knight, then Grace is a knave, and Grace’s statement that “Aurora is a knight and Grace is a knight” is false, which means that Alexander’s statement is false. Therefore, Alexander must be a knave. Grace says that Aurora is a knight and Grace is a knight, but since Alexander is a knave, Grace’s statement that “Grace is a knight” is false, so Grace must be a knave. Avery says that “Grace is a knight if and only if Avery is a knight,” but since Grace is a knave, this statement is false, so Avery must be a knave. Aurora says that “Avery is not a knave,” but since Avery is a knave, this statement is false, so Aurora must be a knave. </think>

<answer> **Alexander is a knave. Grace is a knave. Avery is a knave. Aurora is a knave. (Wrong answer)** </answer>

C²GSPG Answer (Confidence: 0.7306): <think> Let’s analyze the statements one by one and see how they interconnect.

1. Alexander says “Grace is not a knight.”
2. Grace says “Aurora is a knight and Grace is a knight.”
3. Avery says “Grace is a knight if and only if Avery is a knight.”
4. Aurora says “Avery is not a knave.”

The analysis begins by noting a contradiction between Alexander’s and Grace’s statements, leading to the initial conclusion that Alexander is a knave. This implies Grace is not a knight, which in turn makes Avery and Aurora knaves.

However, the reasoning then shifts: if Grace is a knave, Alexander’s statement would be true, making Alexander a knight. This creates a loop.

A significant portion of the reasoning focuses on Avery’s statement under the assumption that Grace is a knave. The logic concludes that for Avery’s biconditional statement (“False if and only if...”) to be true, Avery must be a knave. Since this aligns, his statement is deemed true, which would make Avery a knight. The text explicitly notes this as a contradiction: “There seems to be a contradiction because we concluded both Alexander and Avery as knights, but previously stated Avery is a knave.”

After several cycles of re-evaluation, the reasoning attempts to settle on a final conclusion by testing the consistency of one particular outcome.

To summarize the final proposed path:

- Alexander is a knight (because his statement “Grace is not a knight” is deemed true).
- Grace is a knave (consequently).
- Avery is a knave (because his statement “Grace is a knight if and only if Avery is a knight” becomes *False* \iff *Knave*, which is a false statement, consistent with him being a knave).
- Aurora is a knight (because her statement “Avery is not a knave” is now considered true, despite the previous step concluding Avery is a knave. This reflects the inconsistency in the original text).

</think>

<answer> **Alexander is a knight. Grace is a knave. Avery is a knave. Aurora is a knight. (Wrong answer)** </answer>