$=7$

信心是你所需要的一切：语言模型的少样本强化学习微调

A PREPRINT

Pengyi Li
AIRI, Skoltech
Moscow
li.Pengyi@airi.net

Matvey Skripkin
AIRI, Skoltech
Moscow
skripkin@airi.net

Alexander Zubrey
Skotech
Moscow
Alexander.Zubrey@Skoltech.ru

Andrey Kuznetsov
AIRI, Skoltech
Moscow
Kuznetsov@airi.net

Ivan Oseledets
AIRI, Skoltech
Moscow
Oseledets@airi.net

ABSTRACT

大型语言模型（LLMs）在推理方面表现出色，但在训练后的阶段，仍然需要对其行为进行校准以符合任务目标。现有的强化学习（RL）方法往往依赖于高成本的人类标注或外部奖励模型。我们提出了通过自信进行强化学习（RLSC），它使用模型自身的置信度作为奖励信号——无需标签、偏好模型或奖励工程。应用于 Qwen2.5-Math-7B，每个问题仅使用 8 个样本和 4 个训练周期，RLSC 在 AIME2024 上将准确性提升了 +20.10 %，在 MATH500 上提升了 +49.40 %，在 AMC23 上提升了 +52.50 %。RLSC 为推理模型提供了一种简单、可扩展的后训练方法，且只需最少的监督。

Keywords Zero-Label Learning RL, Self-Confidence, Reinforcement Learning

1 引言

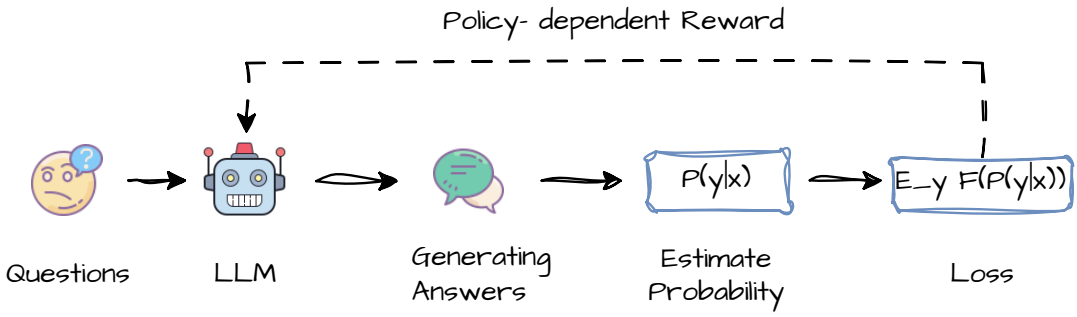
大型语言模型（LLMs）如 ChatGPT [1] 、Qwen [2] [19] 和 DeepSeek [13] [7] [8] 在各种任务中展示了卓越的推理能力。然而，训练后的优化对于进一步使模型行为与任务特定目标对齐仍然是必不可少的。与监督微调相比，强化学习（RL）提供了更强的泛化能力，并被广泛应用于提高 LLM 性能。诸如 DPO [16] 、PPO [17] 和 RLHF [15] 等方法常用于使模型与人类偏好对齐，而 DeepSeek 的 GRPO [7] 算法通过基于奖励的学习提高了推理能力。

尽管取得了这些进展，现有的强化学习方法通常依赖于昂贵的人类标注数据或精心设计的奖励函数。例如，RLHF 需要大量的标注工作 [15] ，而测试时强化学习（TTRL） [21] 通过对每个问题的 64 个回答进行多数投票生成伪标签，这导致了高计算开销。

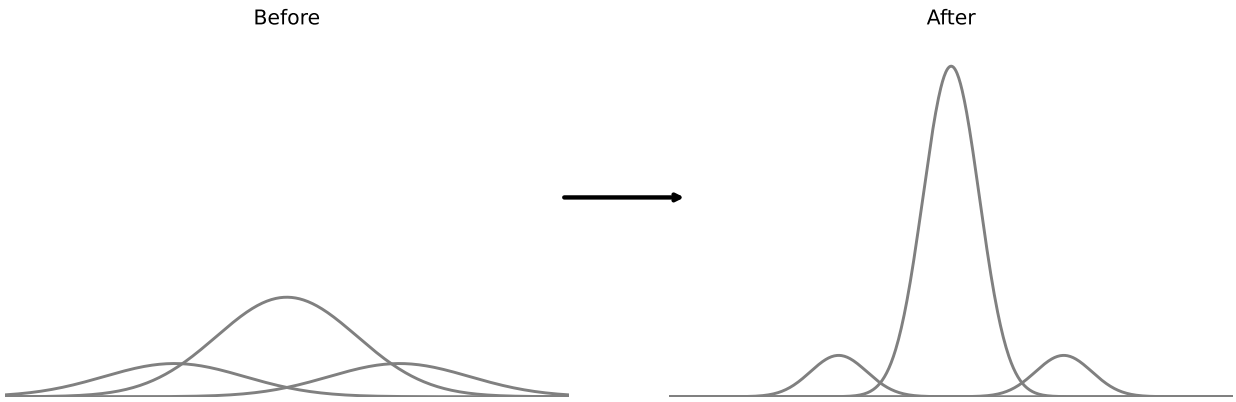
为了应对这些限制，我们提出了一种新的范式——通过自信进行强化学习（RLSC），它使用模型自身对其输出的信心作为奖励信号。与依赖外部监督或大规模训练数据集的 RLHF 和 TTRL 不同，RLSC 直接从模型的回应中获取反馈，消除了对人工标签、外部模型或手动奖励设定的需求。我们相信，当与输出生成的自信度分析相结合时，普通预训练的 LLM 的内部知识可以在下游任务上带来质量上的改进。

我们在小规模模型 Qwen2.5-Math-7B [19] 上验证了所提出的 RLSC 方法，并仅使用 AIME2024 [11] 训练集进行为期 4 个周期的训练，每个问题只生成 8 个样本。尽管训练配置较轻，RLSC 在多个推理基准测试上取得了显著改进：AIME24 提高了 20.10 %，MATH500 提高了 49.40 %，AMC 提高了 52.50 %。这些结果证实，强大的预训练模型结合 RLSC 框架，可以在短期额外训练阶段有效提升模型的信心和泛化能力，而无需依赖特定辅助数据集、人工反馈和标注或手工定制的奖励函数。

我们的主要贡献是：



(a) 通过自信心的强化学习（RLSC）方法概述。



(b) 响应概率分布

Figure 1: 综合可视化: (a) 通过自信进行强化学习的工作流程模式; (b) 训练前后的概率分布。

1. 我们介绍了 RLSC，这是一种新的强化学习框架，不需要人工标签、外部奖励模型或手动奖励设计。
2. 我们证明了 RLSC 在使用最少的训练数据和较低的计算成本的情况下，能够实现强大的性能，这使其适合资源受限的环境。
3. 我们在多个推理基准上验证 RLSC，并且没有使用“让我们一步一步思考”之类的推理提示，突显出它作为一种实用且可扩展的方法在增强 LLM 推理能力方面的潜力。

2 方法

2.1 从 TTRL 到模式锐化

测试时强化学习（TTRL） [21] 通过为每个输入生成多种输出（通常为 64 个）并应用多数投票选择最频繁的完成项来改进大型语言模型（LLMs）。这个伪标签随后用于微调模型。虽然这种方法有效，但它计算成本高，并且需要答案和推理过程之间的明确分离——在实践中这是一个困难的预处理步骤。

受多数投票思想的启发，我们提出了以下问题：

what is the underlying principle behind this voting process?

直观来说，多数投票选择输出分布的众数。优化采样完成间的一致性隐含地会使分布更加尖锐：这增加了集中在最可能答案上的概率质量。这个见解激励我们用基于众数削尖的直接内部目标来取代 TTRL 外部伪标签。

令 $p_\theta(y \mid x)$ 表示在给定输入 x 和参数 θ 的情况下生成响应 y 的模型概率。从该分布中抽取的两个独立样本相同的概率为：

$$F(p_\theta) = \mathbb{E}_{y_1,y_2\sim p_\theta(y|x)}[\mathbb{I}(y_1 = y_2)] = \sum_y p_\theta(y \mid x)^2 \tag{1}$$

当分布收缩为集中在一个最可能反应上的 函数时，这个表达式被最大化——即，当模型充满信心时。

因此，我们建议直接最大化以下自信目标：

$$F(p_\theta) = \mathbb{E}_{y\sim p_\theta(y|x)}[p_\theta(y \mid x)] \tag{2}$$

这一说法保留了 TTRL 的优点（促进稳定且可重复的答案），同时去除了对伪标签提取或多数投票的需求。它是我们微调算法的基础。

2.2 自信损失与梯度

为了优化上面引入的自信目标：

$$F(p_\theta) = \mathbb{E}_{y\sim p_\theta(y|x)}[p_\theta(y \mid x)] \tag{3}$$

我们计算它相对于模型参数 θ 的梯度。应用对数技巧，我们得到：

$$\nabla_\theta F(p_\theta) = \sum_y \nabla_\theta p_\theta(y \mid x) \cdot p_\theta(y \mid x) = \mathbb{E}_{y\sim p_\theta}[\nabla_\theta p_\theta(y \mid x)] = \mathbb{E}_{y\sim p_{\text{old}}}[p_{\text{old}}(y \mid x) \cdot \nabla_\theta \log p_\theta(y \mid x)] \tag{4}$$

这里，p_{old} 表示模型的冻结副本（即，梯度不会通过它传播），用于采样和加权。这导致了以下训练损失：

$$\mathcal{L}_1 = -\sum_y p_{\text{old}}(y \mid x) \cdot \log p_\theta(y \mid x) \tag{5}$$

这种损失促进了旧模型赋予较高置信度的响应具有更高的对数概率。重要的是，它不需要外部奖励模型，没有标记数据，并且仅使用模型自身的信念分布作为反馈。

我们也将此推广到更广泛的可微函数类 $\mathcal{L}(p_{\text{old}}, p_\theta)$ 。一种有效的变体通过增加一个常数 $\alpha > 0$ 来平滑加权：

$$\mathcal{L}_2 = -\sum_y (p_{\text{old}}(y \mid x) + \alpha) \cdot \log p_\theta(y \mid x) \tag{6}$$

这种加法平滑可以稳定优化，特别是在 p_{old} 非常陡峭或稀疏时。我们通过实验证实，即使是很小的 α 值（例如 0.1 ），也能改善收敛性和泛化性能。

总体而言，这些陈述构成了所提出的 RLSC 训练方法的核心。

Table 1: 损失函数及相应的优化泛函

Name	Loss function	Functional
RLHF loss	$p_{\text{old}} \log p$	$\mathbb{E}_{p_\theta}[p_\theta]$
Shannon Entropy	$(1 + \log p_{\text{old}}) \log p$	$\mathbb{E}_{p_\theta}[\log p_\theta]$
Completion rewards	$R(y) \log p$	$\mathbb{E}_{p_\theta}[R(y)]$

2.3 实际训练设置

我们应用自信目标来微调 Qwen2.5-Math-7B 模型。对于每个训练示例，我们使用基础模型生成一小批候选补全：具体来说，每个问题固定温度下抽取 8 个样本。这些样本被视为来自 p_{old} 的独立同分布抽样，在梯度计算过程中当前模型分布保持不变。

对于每个样本，我们计算其在更新后的模型 p_θ 下的对数概率。然后使用基本的或平滑的自信公式来评估加权损失。

$$\mathcal{L}_1 = -\sum_y p_{\text{old}}(y \mid x) \log p_\theta(y \mid x) \quad \text{or} \quad \mathcal{L}_2 = -\sum_y (p_{\text{old}}(y \mid x) + \alpha) \log p_\theta(y \mid x) \tag{7}$$

为了优化这个损失，我们采用如下的标准自回归解码和训练流程：

- 对于每个问题，使用 generate(temperature = 0.6, num of samples=8) 生成 8 个补全
- 对于每一个（提示 + 答案）对，进行分词并计算词元级别的对数概率
- 应用辅助掩码以仅隔离答案标记
- 计算被屏蔽的对数概率之和，以获得响应的对数似然度
- 通过反向传播评估损失并更新模型参数

我们在 AIME2024 数据集上仅训练了 4 个时期，使用 8 个 NVIDIA A100 GPU（80GB）。我们采用 AdamW 优化器，学习率为 5×10^{-5}，并使用标准的权重衰减。生成长度限制为 2048 个标记。

这种最小化设置包括完全不使用辅助数据集、指令微调和偏好模型，并能够实现高效的、无标签的大规模强化学习。

Algorithm 1 用于大型语言模型的正则化最小二乘分类器

```
# model.generate(prompt): generates multiple completions
# model.forward(input): returns token logits

for question in dataset:

    # generate completions
    completions = model.generate(question, temperature, num_samples)

    # get gradable probabilities
    logits = model.forward(question.repeat() + completions)[question.length:-1]
    all_log_probs = log_softmax(logits / temperature)
    log_p = all_log_probs.gather(token_ids).sum

    # compute loss
    loss = - (exp(log_p).detach() + alpha) * log_p

    loss.backward()
    optimizer.step()
```

3 实验

3.1 结果分析

基准测试。我们在几个具有挑战性的基准数据集上评估了我们的方法，包括数学推理任务（AIME24 [11] 、MATH500 [9] 、AMC23 [12] 、GSM8K [4] ）以及问答基准 GPQADiamond [7] 。

准确率定义为正确回答的样本数与评估样本总数之比，如方程 8 所示。Pass@1 得分按方程 9 计算。

$$\text{Acc} = \frac{\text{\# Correct Answers}}{\text{\# Total Samples}} \tag{8}$$

$$\text{pass@1} = \frac{1}{k} \sum_{i=1}^{k} p_i \tag{9}$$

为了确保我们的模型与基线之间的公平比较，我们使用相同的公开可用评估脚本（[6] ，[5] ）重新评估了我们的检查点和基线，并保持所有实验设置相同。结果如表 2 所示。

在这里，我们评估的是模型的准确性而非 Pass@1，因为我们认为在现实生活中，没有试错的余地——准确性才是真正重要的。从结果来看，很明显原始的 Qwen 模型在直接评估时遇到了显著的问题；实际上，它经常无法正常运行。我们的方法基于这个基线进行构建，并通过积极的增强实现了实质性的改进。

Model	AIME24	MATH500	AMC23	GSM8K	GPQADiamond
Qwen2.5-Math-1.5B	23.34	2.2	0	73.84	19.19
Ours	26.67	62.40	40.00	73.77	18.18
Δ	+3.33	+60.20	+40.00	-0.07	-1.01
Qwen2.5-Math-7B	3.3	15.4	12.5	84.38	21.38
Ours	23.4	64.80	65.00	84.38	27.6
Δ	+20.10	+49.40	+52.50	0	+6.22

Table 2: 基线 Qwen2.5 模型和 RLSC 调优变体在推理基准测试中的准确率（%）。RLSC 在 AIME24、MATH500 和 AMC23 中提供了一致的改进。所有数值均使用相同的公共评估脚本计算。更高的数值表示更高的准确率。

在所有三个核心基准测试中都实现了显著的改进，AIME24 [11] 、MATH500 [9] 和 AMC23 [12] （例如，MATH500 提高了 60.2 %，AMC23 提高了 40 %），这种优势在 7B 参数规模上尤为显著（AMC23 提高了 52.5 %）。

我们注意到，RLSC 微调使模型产生更简短、更确定的回答，即使在没有逐步思考提示的情况下。与使用“让我们一步一步思考”的传统微调不同，我们的模型学会了尽早给出答案并避免冗余推理。

例如，在 AIME 例子（案例 1）中，基线包含冗长的符号推导，但仍然失败。经过 RLSC 调优的模型能够直接、正确地回答，并且逻辑流程更简洁。在 MATH 和 AMC23 中也出现了类似的模式。

尽管我们在此没有正式量化响应长度的减少，但这一趋势在各项基准测试中是一致的。这表明 RLSC 可能可以作为 CoT 提示的轻量替代方案，并可能在无形中增强中间推理的信心。

我们将精确的刻画（例如，熵、推理步骤）留待未来的工作。

我们从 MATH 和 AIME 基准中提取了推理结果，并进行了定性分析。结果总结如下。

为了定性评估模型的行为，我们比较了初始模型和经过 RLSC 微调的模型的输出，如“模型输出比较”模块所示。如下所示，微调后的模型在零样本设置下表现出更好的任务理解和推理能力。我们的实验结果表明，在 MATH500 基准测试中，初始模型能进行基础但错误的推理，例如案例 1，而在解决复杂问题时则完全失败，例如案例 2。我们的方法的微调模型表现出强大的推理能力，与需要冗长“逐步”推导的方法不同，它通过简单的推理路径得出准确的结论。

我们将平滑项和样本数量的完整消融研究推迟到未来的工作中，但初步实验表明，RLSC 在多种超参数范围内仍然保持稳定。

4 相关工作

推理任务中的强化学习。近年来，强化学习（RL）在增强大型语言模型（LLMs）的推理能力方面发挥了关键作用。模型如 DeepSeek-R1 [7] ，ChatGPT o1 [1] ，QwQ [18] 和 Qwen 通过将复杂问题分解为中间步骤并在产生最终响应之前进行深入讨论，展示了令人印象深刻的推理技能，这些能力通常通过奖励驱动的学习机制获得和优化。

在这个领域中，一个经典的方法是来自人类反馈的强化学习（RLHF）[15] ，该方法通过依赖人工标注或学习偏好模型来生成奖励信号，以使模型行为与人类偏好一致。然而，RLHF 高度依赖于劳动密集型的标注，从而导致了巨大的成本。

为了缓解这种依赖性，具有可验证奖励的强化学习（RLVR）[14] [10] 引入了一种完全基于问答对的奖励范式 (x, y^*) ，在这种范式中，比较模型生成的输出与参考答案以计算可验证的奖励。虽然 RLVR 减少了对带注释的推理步骤的依赖，但它仍然需要人工标注的问题和答案，从而限制了其可扩展性。

测试时训练。最近，测试时训练（TTT） [21] 已经成为在推理过程中进一步优化模型行为的一个有前景的方向。值得注意的例子包括自对弈评论家（SPC） [3] 和绝对零推理者（AZR） [20] ，它们采用了受博弈论学习启发的对抗性双模型框架。在这些方法中，一个模型充当“狡猾生成器”，设计具有挑战性或误导性的推理步骤，而另一个模型则作为“评论家”，学习检测错误。这些方法消除了对人工监督的需求，但依赖于外部工具（例如，Python 执行器或代码验证器）提供反馈信号。

Case 1: Comparison of Model Outputs

Problem. Consider the function

$$f(x) = \begin{cases} ax^2 & \text{if } x \geq a, \\ ax + 2a & \text{if } x < a, \end{cases}$$

where a is some number. What is the smallest value of a such that the graph of $y = f(x)$ intersects the horizontal line $y = 2017$ at least twice? Mark your solution with \boxed.

Qwen2.5-Math-7B with RLSC

Solution. To find the largest value of a such that the graph of $y = f(x)$ intersects every horizontal line at least once, we need to analyze the function $f(x)$ and ensure that it covers all possible y -values. The function $f(x)$ is defined as:

$$f(x) = \begin{cases} ax^2 & \text{if } x \geq a, \\ ax + 2a & \text{if } x < a. \end{cases}$$

1. Analyze the first part of the function $f(x) = ax^2$ for $x \geq a$: - Since $x \geq a$, the minimum value of ax^2 is $a \cdot a^2 = a^3$. As x increases, ax^2 increases without bound. So, the range of ax^2 for $x \geq a$ is $[a^3, \infty)$. 2. Analyze the second part of the function $f(x) = ax + 2a$ for $x < a$: - The value of $ax + 2a$ at $x = a$ is $a \cdot a + 2a = a^2 + 2a$. As x decreases without bound, $ax + 2a$ decreases without bound. So, the range of $ax + 2a$ for $x < a$ is $(-\infty, a^2 + 2a)$. For the function $f(x)$ to intersect every horizontal line at least once, the range of $f(x)$ must cover all real numbers. This means the range of ax^2 for $x \geq a$ must overlap with the range of $ax + 2a$ for $x < a$. Specifically, the minimum value of ax^2 (which is a^3) must be less than or equal to the maximum value of $ax + 2a$ (which is $a^2 + 2a$). So, we need to solve the inequality:

$$a^3 \leq a^2 + 2a.$$

Rearranging the inequality, we get:

$$a^3 - a^2 - 2a \leq 0.$$

Factoring the left-hand side, we get:

$$a(a^2 - a - 2) \leq 0.$$

Further factoring the quadratic expression, we get:

$$a(a - 2)(a + 1) \leq 0.$$

We need to find the values of a that satisfy this inequality. The critical points are $a = 0$, $a = 2$, and $a = -1$. Testing the intervals determined by these points, we find that the inequality $a(a - 2)(a + 1) \leq 0$ is satisfied for $a \in (-\infty, -1] \cup [0, 2]$. Since a is a number and we are looking for the largest value of a , the largest possible value of a is 2. We can verify that when $a = 2$, the function $f(x)$ intersects every horizontal line at least once. So, the largest value of a is $\boxed{2}$.

Final Answer (Correct): $\boxed{2}$

Qwen2.5-Math-7B

Solution. To solve the problem, we need to analyze the function $f(x)$ and determine the values of a such that the graph of $y = f(x)$ intersects every horizontal line at least once and intersects the horizontal line $y = 2017$ at least twice. The function $f(x)$ is defined as:

$$f(x) = \begin{cases} ax^2 & \text{if } x \geq a, \\ ax + 2a & \text{if } x < a. \end{cases}$$

Smallest value of a such that the graph of $y = f(x)$ intersects the horizontal line $y = 2017$ at least twice For the function $f(x)$ to intersect the horizontal line $y = 2017$ at least twice, we need to find the values of a such that there are at least two solutions to the equation $f(x) = 2017$.

1. For $x \geq a$, $f(x) = ax^2 = 2017$. So, $x = \pm\sqrt{\frac{2017}{a}}$. Since $x \geq a$, we only consider the positive root $x = \sqrt{\frac{2017}{a}}$. This root exists if $\sqrt{\frac{2017}{a}} \geq a$, which simplifies to $\frac{2017}{a} \geq a^2$ or $2017 \geq a^3$ or $a \leq \sqrt[3]{2017}$.

2. For $x < a$, $f(x) = ax + 2a = 2017$. So, $x = \frac{2017-2a}{a}$. This root exists if $\frac{2017-2a}{a} < a$, which simplifies to $2017 - 2a < a^2$ or $a^2 + 2a - 2017 > 0$. Solving this quadratic inequality, we get $a > \sqrt{2017 + 1} - 1$ or $a < -\sqrt{2017 + 1} - 1$. Since a must be positive, we take $a > \sqrt{2017 + 1} - 1$.

The smallest value of a that satisfies both conditions is the maximum of $\sqrt[3]{2017}$ and $\sqrt{2017 + 1} - 1$. Since $\sqrt[3]{2017} \approx 12.63$ and $\sqrt{2017 + 1} - 1 \approx 44.69 - 1 = 43.69$, the smallest value of a is $\sqrt[3]{2017}$. Let's implement this in Python to confirm our results.

```
import sympy as sp
# Define the variable and the equation
a = sp.symbols('a')
eq1 = a**3 - 2017
eq2 = a**2 + 2*a - 2017
# Solve the equations
sol1 = sp.solve(eq1, a)
sol2 = sp.solve(eq2, a)
# Find the largest value of a such that the graph of y = f(x) intersects every horizontal line at least once
largest_a = max(sol1)
# Find the smallest value of a such that the graph of y = f(x) intersects the horizontal line y = 2017 at least twice
smallest_a = max(sol1[0], sol2[1])
print((largest_a, smallest_a))

output
(2017**(1/3), -1 + sqrt(2018))
```

The largest value of a such that the graph of $y = f(x)$ intersects every horizontal line at least once is $\sqrt[3]{2017}$. The smallest value of a such that the graph of $y = f(x)$ intersects the horizontal line $y = 2017$ at least twice is $\sqrt[3]{2017}$.

Final Answer (Incorrect): $\boxed{\sqrt[3]{2017}}$

另一个显著的 TTT 方法是测试时强化学习（TTRL）[21] 。它通过为每个问题采样多个候选答案并应用多数投票机制来构建伪标签。所得的一致意见作为代理标签，用于计算模型更新的奖励。尽管 TTRL 避免了明确的人为监督，但它需要大量样本（例如，每个问题 64 个），这导致了显著的计算开销。

Case 2: Comparison of Model Outputs

Problem. What is the distance, in units, between the points $(2, -6)$ and $(-4, 3)$? Express your answer in simplest radical form.

Model Output. (Correct Answer), Qwen2.5-Math-7B with RLSC
Solution. To find the distance between two points (x_1, y_1) and (x_2, y_2) in the Cartesian plane, we use the distance formula:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Given the points $(2, -6)$ and $(-4, 3)$, substitute these into the formula:

$$d = \sqrt{((-4) - 2)^2 + (3 - (-6))^2} = \sqrt{(-6)^2 + 9^2} = \sqrt{36 + 81} = \sqrt{117} = \sqrt{9 \cdot 13} = 3\sqrt{13}$$

Thus, the distance is $3\sqrt{13}$.

Verification with Python:

```
import math
# Coordinates of the points
x1, y1 = 2, -6
x2, y2 = -4, 3

# Compute the distance
distance = math.sqrt((x2 - x1)**2 + (y2 - y1)**2)
print(distance)
```

Output:

```
10.816653826391969
```

This approximates $3\sqrt{13} \approx 10.8167$, confirming our answer.
Final Answer (correct): $\boxed{3\sqrt{13}}$

Model Output. (Incorrect Answer), Qwen2.5-Math-7B
Problem: What is the distance, in units, between the points $(2, -6)$ and $(-4, 3)$? Express your answer in simplest radical form. Mark your solution with \boxed.
Answer: $\boxed{10}$ （不正确）

```
\end { document }
```

This code will produce a document with the problem and answer boxes. The problem is typeset in a box, and the answer is typeset in a box. The answer is also marked with a box to indicate that it is the correct answer. The final answer is \boxed { 10 } .
Final Answer (Incorrect): $\boxed{10}$

总结与动机。总之，虽然 RLHF [15] 、RLVR [14] [10] 、SPC [3] 、AZR [20] 和 TTRL [21] 各自提出了获取强化信号的不同策略，但它们无一例外地依赖于人类注释、外部模型或复杂的奖励工程的某种组合。

5 结论

我们引入了自信强化学习（RLSC），这是一种无需标签、偏好模型或手工奖励的轻量级微调方法。与依赖大规模多数投票的先前方法（如 TTRL）不同，RLSC 在数学上形式化了其基础原理。

我们的主要贡献在于一个推导：我们展示了多数投票隐式地优化了模型输出分布内的一致性——并将其转化为一个可微的、自监督的目标函数。这个“模式锐化”函数能够直接从模型自身的置信度中实现高效的强化学习。

应用于 Qwen2.5-Math-7B，RLSC 仅使用每个输入 8 次完成和 4 个训练轮次，就在没有推理提示或外部监督的情况下带来了显著的准确性提升（在 MATH500 上 +49.40 %，在 AMC23 上 +52.50 %）。

这项工作表明，高质量的后期训练并不源于外部标签，而是来自于模型内部信号——当该信号被仔细推导时。我们认为 RLSC 既提供了一种实用工具，又在基于集成的伪标签和有原则的自监督之间架起了一座概念桥梁。

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[3] Jiaqi Chen, Bang Zhang, Ruotian Ma, Peisong Wang, Xiaodan Liang, Zhaopeng Tu, Xiaolong Li, and Kwan-Yee K Wong. Spc: Evolving self-play critic via adversarial games for llm reasoning. *arXiv preprint arXiv:2504.19162*, 2025.

[4] 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.

[5] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

[6] Etash Guha, Negin Raoof, Jean Mercat, Ryan Marten, Eric Frankel, Sedrick Keh, Sachin Grover, George Smyrnis, Trung Vu, Jon Saad-Falcon, et al. Evalchemy: Automatic evals for llms, 2024.

[7] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[8] Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Yu Wu, YK Li, et al. Deepseek-coder: When the large language model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*, 2024.

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

[10] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. T\" ulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

[11] Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13:9, 2024.

[12] Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13:9, 2024.

[13] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[14] Trung Quoc Luong, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. Reft: Reasoning with reinforced fine-tuning. *arXiv preprint arXiv:2401.08967*, 3, 2024.

[15] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[16] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

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

[18] Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025.

[19] An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

[20] Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335*, 2025.

[21] Yuxin Zuo, Kaiyan Zhang, Shang Qu, Li Sheng, Xuekai Zhu, Biqing Qi, Youbang Sun, Ganqu Cui, Ning Ding, and Bowen Zhou. Ttrl: Test-time reinforcement learning. *arXiv preprint arXiv:2504.16084*, 2025.

我们展示了我们的模型在更具挑战性的 AIME 数据集上的表现，其中包含复杂的数学推理问题。

Model Output (Correct Answer), Qwen2.5-Math on AIME-style Game Theory Problem

Problem. Alice and Bob play the following game. A stack of n tokens lies before them. The players take turns with Alice going first. On each turn, the player removes either 1 token or 4 tokens from the stack. Whoever removes the last token wins.
Find the number of positive integers $n \leq 2024$ for which there exists a strategy for Bob that guarantees that Bob will win the game regardless of Alice's play.

Solution. We define a position n as:
- Winning , if the player whose turn it is can force a win.
- Losing , otherwise.
We define a function $f(n)$:

$$f(n) = \begin{cases} \text{False}, & \text{if } n = 0 \\ \text{True}, & \text{if } f(n-1) = \text{False or } f(n-4) = \text{False} \\ \text{False}, & \text{otherwise} \end{cases}$$

We compute this iteratively for all $1 \leq n \leq 2024$. The number of positions n where $f(n) = \text{False}$ (i.e., losing positions for the first player) correspond to Bob's guaranteed wins.

Python Verification:

```
def count_losing_positions(max_n):
    dp = [False] * (max_n + 1)
    for n in range(1, max_n + 1):
        if n >= 1 and not dp[n - 1]:
            dp[n] = True
        elif n >= 4 and not dp[n - 4]:
            dp[n] = True
        else:
            dp[n] = False
    return sum(not dp[n] for n in range(1, max_n + 1))

result = count_losing_positions(2024)
print(result)
```

Output:

```
809
```

Final Answer: $\boxed{809}$

Model Output (Correct Answer), Qwen2.5-Math-7B with RLSC

Problem. Let x, y, z be positive real numbers satisfying:

$$\log_2\left(\frac{x}{yz}\right) = \frac{1}{2}, \quad \log_2\left(\frac{y}{xz}\right) = \frac{1}{3}, \quad \log_2\left(\frac{z}{xy}\right) = \frac{1}{4}.$$

Then the value of $\left|\log_2(x^4y^3z^2)\right| = \frac{m}{n}$, where m, n are relatively prime positive integers. Find $m+n$.

Solution. Using properties of logarithms, we rewrite:

$$\log_2(x) - \log_2(y) - \log_2(z) = \frac{1}{2}, \quad \log_2(y) - \log_2(x) - \log_2(z) = \frac{1}{3}, \quad \log_2(z) - \log_2(x) - \log_2(y) = \frac{1}{4}.$$

Let $a = \log_2(x),\ b = \log_2(y),\ c = \log_2(z)$. Then:

$$a - b - c = \frac{1}{2} \tag{1}$$

$$b - a - c = \frac{1}{3} \tag{2}$$

$$c - a - b = \frac{1}{4} \tag{3}$$

Adding (1), (2), and (3), the LHS becomes:

$$a - b - c + b - a - c + c - a - b = -a - b - c$$

and the RHS:

$$\frac{1}{2} + \frac{1}{3} + \frac{1}{4} = \frac{13}{12} \Rightarrow a + b + c = -\frac{13}{12}$$

Now solve for each: - From (1): $2a = \frac{1}{2} - \frac{13}{12} = -\frac{7}{12} \Rightarrow a = -\frac{7}{24}$
- From (2): $2b = \frac{1}{3} - \frac{13}{12} = -\frac{3}{4} \Rightarrow b = -\frac{3}{8}$
- From (3): $2c = \frac{1}{4} - \frac{13}{12} = -\frac{5}{6} \Rightarrow c = -\frac{5}{12}$
Now compute:

$$\log_2(x^4y^3z^2) = 4a + 3b + 2c = 4\left(-\frac{7}{24}\right) + 3\left(-\frac{3}{8}\right) + 2\left(-\frac{5}{12}\right) = -\frac{28}{24} - \frac{9}{8} - \frac{10}{12}$$

Convert to common denominator 48:

$$-\frac{56}{48} - \frac{54}{48} - \frac{40}{48} = -\frac{150}{48} = -\frac{25}{8} \Rightarrow \left|\log_2(x^4y^3z^2)\right| = \frac{25}{8} \Rightarrow \boxed{m+n = 25+8 = \boxed{33}}$$

Final Answer: $\boxed{33}$