

# 目 录

<b>1</b>	<b>引言</b>	<b>5</b>
<b>2</b>	<b>GRPO 算法简介</b>	<b>5</b>
2.1	动机	5
2.2	workflows	5
2.3	GRPO 算法的关键公式	5
<b>3</b>	<b>DeepSeek 训练的基本原理</b>	<b>6</b>
3.1	奖励函数构建	6
3.1.1	奖励构建范式	6
3.2	Zero 训练范式	6
3.3	模型训练中的自进化现象	6
3.4	R1 的技术框架	7
3.4.1	冷启动优化机制	7
3.4.2	语义一致性的引入	7
3.4.3	抑制性采样策略	7
3.5	替代方法可行性分析	7
3.5.1	过程奖励模型的局限性	7
3.5.2	蒙特卡洛树搜索适配挑战	7
<b>4</b>	<b>本文的复现思路</b>	<b>7</b>
4.1	冷启动数据集的构建	7
4.2	奖励函数设计思想	8
4.3	SFT 的设计	9
4.4	超参数设计	9
4.5	总体流程说明	10
4.6	实验环境说明	10
<b>5</b>	<b>结果展示</b>	<b>10</b>
5.1	生成结果分析	10
5.2	基于 R1-Zero 的结论	11
5.2.1	强化学习带来的思维强化	11
5.2.2	数据集评测结果	11
5.3	SFT 的引入与消融	11
5.3.1	SFT 带来的直接增益	11
5.3.2	SFT 后的 RL	11
5.4	RL 引起的思维涌现的报告	11
<b>6</b>	<b>讨论</b>	<b>12</b>
<b>A</b>	<b>冷启动数据集的构建模板</b>	<b>13</b>
<b>B</b>	<b>Case Study</b>	<b>13</b>
B.1	SFT 数据集	13
B.2	推理涌现	14



# 1 引言

我的选题为强化学习方向的实践题的自拟题目。报告内容为：针对 DeepSeek-R1<sup>[1]</sup>的猜测性复现，之所以选择该题的原因是：DeepSeek-R1 的出色表现显著的影响了当今自然语言处理的范式，引起了国内外的显著反响，但是，尽管 DS 的权重已被开源，但是其训练的具体过程并没有得到公开，尽管在技术报告里，DS 官方简述了他们训练的 pipeline 以及实验的流程，但是他们究竟是如何实现的，奖励函数的设计细节，冷启动数据的构建方式，基于小模型的进一步尝试并没有在论文里得到充分的展开，因此，我决定以本课程为契机，探索 DS 的一种构建方式。本大作业将围绕以下内容展开：

1. GRPO 算法简介
2. DeepSeek 训练的基本原理
3. 本文的复现思路
4. 结果展示
5. 讨论

## 2 GRPO 算法简介

### 2.1 动机

在传统的强化学习中，模型会根据环境给出的奖励信号来调整自己的行为。通常，这会涉及一个额外的模型，用来评估当前策略的好坏。然而，批评模型的训练既复杂又耗费计算资源。GRPO<sup>[2]</sup>的核心思想是简化这个过程：它不需要批评模型，而是通过组内相对奖励来优化策略模型。具体来说，GRPO 会从当前策略中采样一组输出，然后根据这些输出的相对表现来调整策略，而不是依赖一个单独的批评模型。

### 2.2 workflow

GRPO 的工作流程可以分为以下几个步骤：

1. 采样一组输出：对于每个问题，GRPO 会从当前策略中采样一组输出，这些输出可以看作是模型对同一个问题的不同“尝试”。
2. 计算组内相对奖励：对这组输出进行评分，计算每个输出的奖励（例如，答案是否正确、推理过程是否合理），然后，GRPO 会计算每个输出的相对优势，即它的奖励相对于组内其他输出的表现如何。
3. 优化策略模型：根据这些相对优势，GRPO 会调整策略模型，使得表现较好的输出更有可能被生成，而表现较差的输出被抑制，这个过程通过数学公式（如梯度上升）来实现，逐步优化模型的策略。

### 2.3 GRPO 算法的关键公式

GRPO 算法的核心目标函数如下：

$$J_{\text{GRPO}}(\theta) = \mathbb{E}_{q \sim P(Q), \{o_i\}_{i=1}^G \sim \pi_{\text{old}}(O|q)} \left[ \frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} (r_{i,t}(\theta) \hat{A}_{i,t} - \beta D_{\text{KL}}(\pi_{\theta} || \pi_{\text{ref}})) \right] \quad (1)$$

其中：

- $G$  是采样的输出组的数量，即对每个问题  $q$ ，从策略模型  $\pi_{\theta}$  中采样  $G$  个输出  $o_1, o_2, \dots, o_G$ ；
- $r_{i,t}(\theta)$  表示第  $i$  个输出  $o_i$  在时间步  $t$  的策略比率，定义为：

$$r_{i,t}(\theta) = \frac{\pi_{\theta}(o_{i,t}|q, o_{i,<t})}{\pi_{\text{old}}(o_{i,t}|q, o_{i,<t})} \quad (2)$$

- $\hat{A}_{i,t}$  是第  $i$  个输出  $o_i$  在时间步  $t$  的相对优势值 (Relative Advantage)，通过组内奖励计算：

$$\hat{A}_{i,t} = \frac{r_i - \text{mean}(r)}{\text{std}(r)} \quad (3)$$

其中  $r_i$  为第  $i$  个输出的奖励， $\text{mean}(r)$  和  $\text{std}(r)$  分别表示组内奖励的均值和标准差；

- $D_{\text{KL}}(\pi_{\theta}||\pi_{\text{ref}})$  是策略分布  $\pi_{\theta}$  和参考策略  $\pi_{\text{ref}}$  之间的 KL 散度，用作正则化，定义为：

$$D_{\text{KL}}(\pi_{\theta}||\pi_{\text{ref}}) = \frac{\pi_{\text{ref}}(o_{i,t}|q, o_{i,<t})}{\pi_{\theta}(o_{i,t}|q, o_{i,<t})} - \log \left( \frac{\pi_{\text{ref}}(o_{i,t}|q, o_{i,<t})}{\pi_{\theta}(o_{i,t}|q, o_{i,<t})} \right) - 1 \quad (4)$$

- $\beta$  是 KL 散度的正则化系数，用于平衡策略更新和参考策略之间的偏差。

### 3 DeepSeek 训练的基本原理

#### 3.1 奖励函数构建

DeepSeek 模型采用第2节介绍的 GRPO 算法进行策略优化，该算法在平衡探索与开发方面展现出显著优势。与常规方法不同，DeepSeek-R1 创新性地摒弃了过程奖励模型（process reward model）和结果奖励模型（outcome reward model）。如技术报告所述，这种设计能有效避免训练过程中出现的奖励黑客现象（reward hacking），即模型通过非预期方式人为抬高奖励信号的问题。

##### 3.1.1 奖励构建范式

###### • 正确性奖励

- 确定性结果问题：针对数学等具有确定解的问题，建立基于规则的验证机制。要求模型以指定格式提交最终答案，通过正则表达式匹配或符号计算进行验证。
- 代码生成问题：通过预编译测试用例进行自动化验证，利用编译器反馈构建动态奖励信号。

###### • 格式奖励

除准确性维度外，强制要求模型将推理过程置于 `<think>...</think>` 标签内，该结构化约束显著提升了解释性同时便于自动化解析

#### 3.2 Zero 训练范式

DeepSeek-R1-Zero 的训练采用渐进式优化策略，其核心要素包括：

- 最小化指令模板设计（见表1），仅约束输出格式为“推理过程 + 最终答案”的双段结构
- 刻意避免引入内容相关的先验偏见，包括：
  - 不强制要求反思性推理
  - 不预设特定问题解决策略
- 通过这种设计保留模型的自主进化能力，便于在强化学习过程中准确观测策略空间的自然演化

表 1 基础模型指令模板设计

要素	描述
输入格式	问题描述 + 格式约束说明
输出结构	<code>&lt;think&gt;</code> 推理过程 <code>&lt;/think&gt;</code> 答案：最终答案
约束强度	仅强制结构要求，无内容限制

#### 3.3 模型训练中的自进化现象

在分析 DeepSeek-R1 系列模型的训练机制时，技术报告指出了若干值得注意的自组织现象。首先，模型的推理 token 消耗量与强化学习的步数之间呈现近似线性的增长模式，这表明该系统通过扩展测试时计算资源，逐步提升其处理复杂推理任务的能力。训练数据揭示，模型在未显式编程的情况下，自主发展出反思和替代路径探索等高级认知行为。技术报告指出：这些特征并非源于预先设定的启发式规则，而是强化学习环境交互的自然产物。

### 3.4 R1 的技术框架

基于 R1-Zero 的成功，DS 研究团队提出的 R1 架构采用分阶段训练策略，包含三个核心创新点：

#### 3.4.1 冷启动优化机制

与 Zero 不同，R1 引入结构化冷启动数据（约 800k 样本）来缓解强化学习初期的不稳定问题。

#### 3.4.2 语义一致性的引入

训练阶段引入语言一致性奖励机制以应对多语言混合问题（技术报告指出，他们在 Zero 的训练中发现，但是在我们的实验中未出现，因此我们的复现没有考虑）。

#### 3.4.3 抑制性采样策略

在推理导向的强化学习收敛过程中，R1 利用检查点结果收集监督微调（SFT）数据，以支持后续训练阶段。与初始冷启动数据不同，此阶段整合了来自其他领域的数据，旨在提升模型在写作、角色扮演等通用任务中的表现。具体而言，R1 通过拒绝采样从 RL 训练的检查点生成推理提示和轨迹，扩展数据集以包含额外数据，其中部分数据通过生成奖励模型进行评估。为确保输出质量，R1 过滤掉混合语言、冗长释义和代码块的思维链，并对每个提示采样多个响应，仅保留正确的结果，从而收集了约 60 万条推理相关的训练样本。此外，对于非推理任务（如写作、事实问答、自我认知和翻译），R1 复用其 SFT 数据集的一部分，生成思维链以回答问题，最终收集了约 20 万条与推理无关的训练样本。

### 3.5 替代方法可行性分析

#### 3.5.1 过程奖励模型的局限性

尽管 PRM 在理论层面具备优化推理路径的潜力，实际应用面临三重障碍：1) 细粒度步骤验证需要昂贵的专家标注；2) 模型自动标注存在语义漂移风险；3) 奖励黑箱化导致训练稳定性下降。对比实验显示，PRM 方法相比基线 RLHF 方案需要增加 40% 的计算开销，但准确率提升不足 2%。

#### 3.5.2 蒙特卡洛树搜索适配挑战

将 MCTS 引入 NLP 推理任务面临两个根本性限制：1) 解空间维度爆炸导致搜索效率急剧下降；2) 价值函数建模的粒度问题。初步实验表明，即使设置节点扩展限制 ( $k = 5$ )，单次推理耗时仍增加  $7.8\times$ ，而准确率提升仅 0.3%。此外，价值模型的层次化训练需求与语言生成任务的连续性本质存在内在冲突。

## 4 本文的复现思路

鉴于计算资源和时间成本的显著限制，完整复现上述所有实验在现实中是不可行的。具体而言，基于 DeepSeek 的监督微调（SFT）和强化学习（RL）策略至少需要 8 张 H100 显卡进行数月的训练。因此，作为一项探索性研究，我们选择在 Qwen2.5-1.5B-Instruct 模型上复现 R1 的实现流程。此外，考虑到抑制性采样策略的复杂性，以及在本论文撰写期间 DeepSeek 官方遭遇的分布式拒绝服务（DDoS）攻击的技术障碍，我们决定放弃 3.4.3 中讨论的流程，同样也是由于这个原因，我们未基于 V3 构建冷启动数据。同时，出于成本控制的考虑，我们将研究重点集中在数学问题的解决能力上。本文的所有分析和讨论均基于 GSM8K 数据集展开。

### 4.1 冷启动数据集的构建

受限于成本限制，我们仅仅只构建了一个约 7k 的数据集而不是原始论文中的 200k 的数据集，由于 DS 的 API 暂时被禁用，我们基于 ChatGPT-4o 构建我们的微调数据集，需要说明的是，考虑到模型能力的问题和成本问题，该数据集的数据质量应该远低于 DS 的数据集，我们的 prompt 模板可参考附录 A，同时，为了保证速度，我们也设计了一套并行的多线程访问策略，具体可以参考源代码的实现。

表 2 强化学习奖励函数设计

```

# 准确性奖励函数
def correctness_reward_func(prompts, completions, answer, **kwargs):
    """ 基于答案匹配的二值化奖励机制 """
    responses = [completion[0]['content'] for completion in completions]
    extracted_responses = [extract_xml_answer(r) for r in responses]
    return [2.0 if r == a else 0.0 for r, a in zip(extracted_responses, answer)]

# 严格式奖励函数
def strict_format_reward_func(completions, **kwargs):
    """ 基于正则表达式的严格 XML 结构验证 """
    pattern = r"<think>\n.*?\n</think>\n<answer>\n.*?\n</answer>\n\$"
    responses = [completion[0]["content"] for completion in completions]
    matches = [re.match(pattern, r) for r in responses]
    return [0.5 if match else 0.0 for match in matches]

# 渐进式格式奖励函数
def xmlcount_reward_func(completions, **kwargs):
    """ 基于 XML 元素计数的梯度奖励机制 """
    contents = [completion[0]["content"] for completion in completions]
    return [count_xml(c) for c in contents] # count_xml 实现细粒度评分

```

## 4.2 奖励函数设计思想

奖励函数设计是强化学习的核心环节，参考原论文提到的奖励函数设计思想，我们也从格式奖励和准确性奖励两个角度分析代码中的奖励函数设计思想，其更进一步的细节可参考源代码的实现。

**准确性奖励** 通过 `correctness_reward_func` 实现二元判定机制：

- 使用 `extract_xml_answer` 从响应中提取答案内容
- 与标准答案进行精确匹配 (2.0/0.0)
- 包含调试输出模块用于验证提取过程

**格式奖励** 采用多粒度验证策略：

- **严格验证**： `strict_format_reward_func` 要求精确的 XML 结构和换行格式

$$\text{奖励值} = \begin{cases} 0.5 & \text{匹配 } < \text{think} > \backslash n . * ? \backslash n < / \text{think} > \backslash n < \text{answer} > \backslash n . * ? \backslash n < / \text{answer} > \$ \\ 0.0 & \text{其他情况} \end{cases}$$

- **渐进式验证**： `xmlcount_reward_func` 通过：

- 标签对计数检测（每个有效标签对 +0.125）
- 尾部冗余内容惩罚（每字符-0.001）
- 总分值范围 [0, 0.5]

**混合奖励机制** 通过组合不同权重的奖励函数：

- 格式奖励作为基础约束（总权重 0.5）
- 准确性奖励作为核心目标（权重 2.0）
- 整数验证等辅助奖励增强鲁棒性

### 4.3 SFT 的设计

我们的监督微调设计采样全参数微调，训练 10 个 epochs，具体的学习率等超参数设计参考4.4部分的详细说明，数据集的构建参考前文的说明，数据集的细节可参考附录B.1。

### 4.4 超参数设计

本实验的超参数设计如下表3（SFT），4（强化学习）所示：

表 3 监督微调的超参数设计

```
training_args = TrainingArguments(
    output_dir="./qwen_finetuned",
    num_train_epochs=10,
    per_device_train_batch_size=4,
    per_device_eval_batch_size=1,
    gradient_accumulation_steps=4,
    optim="adamw_torch",
    learning_rate=1e-5,
    weight_decay=0.01,
    warmup_ratio=0.03,
    lr_scheduler_type="cosine",
    logging_steps=10,
    evaluation_strategy="epoch",
    save_strategy="epoch",
    save_total_limit=2,
    bf16=True if torch.cuda.is_available() else False,
    report_to=["tensorboard"],
    load_best_model_at_end=True,
    metric_for_best_model="loss",
)
```

表 4 强化学习的超参数设计

```
training_args = GRPOConfig(
    output_dir=output_dir,
    run_name=run_name,
    learning_rate=5e-6,
    adam_beta1=0.9,
    adam_beta2=0.99,
    weight_decay=0.1,
    warmup_ratio=0.1,
    lr_scheduler_type='cosine',
    logging_steps=1,
    bf16=True,
    per_device_train_batch_size=1,
    gradient_accumulation_steps=4,
    num_generations=8,
    max_prompt_length=256,
    max_completion_length=400,
    num_train_epochs=1,
    save_steps=80,
    max_grad_norm=0.1,
    log_on_each_node=False,
)
```

## 4.5 总体流程说明

如表5所示，我们首先安装环境，然后进行监督微调，之后进行 R1-Zero 和 R1 的强化学习，然后进行统计分析，相应的 sh 脚本如下。

表 5 sh 脚本

```
sh requirment.sh
export HF_ENDPOINT=https://hf-mirror.com
huggingface-cli download --resume-download Qwen/Qwen2.5-1.5B-Instruct --local-dir Qwen/Qwen2.5-1.5B-Instruct
git clone https://hf-mirror.com/datasets/openai/gsm8k
huggingface-cli download --resume-download deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B --local-dir
↳ deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B
python ini.py
python sft.py
python rl-zero.py
python rl-sft.py
python test.py
```

## 4.6 实验环境说明

本实验在单卡 A100 40GB 上完成，实验环境为 Linux 系统下的 pytorch 2.5.0，R1-Zero 的训练用时为 22h，R1 的训练用时为 27h，其中 SFT 占比为 4h，测试在 4060 上即可完成，用时 5h。

# 5 结果展示

## 5.1 生成结果分析

在初始阶段，模型的输出通常缺乏指令遵循的能力，甚至在某些情况下仅生成一个不正确的数字，例如“8”。此外，模型偶尔会表现出向特定模板方向生成的倾向，例如输出形如“box{xxx}”的结构。从输出过程来看，R1 的模板遵循能力迅速收敛，而 Zero 的收敛速度相对较慢。通过对 Zero 生成结果的分析，我们发现这可能与模型的能力限制有关：模型在初始阶段并未遵循预期的指令格式。我们对 Zero 出现的现象进行进一步的细粒度的分析，随着强化学习的推进，如表6所示，模型在大约 20 步左右首次发现输出 `<answer></answer>` 标签能够获得格式上的奖励，随后其输出开始显著向这一方向转变。<sup>1</sup>

表 6 初次标签的发现

```
Let the number of students in the sixth-level class be S = 40. The number of students in the fourth-level class
↳ is 4 times the number of students in the sixth-level class, so the number of students in the fourth-level
↳ class is 4 * S = 4 * 40 = 160. The number of students in the seventh-level class is twice the number of
↳ students in the fourth-level class, so the number of students in the seventh-level class is 2 * 160 = 320.
↳ The total number of students Ms. Cole teaches is the sum of the students in all three levels, so it is S + 4
↳ * S + 2 * 4 * S = 40 + 160 + 320 = 520.<answer>520</answer>
```

随着训练的进一步进行，模型开始逐渐了解到，正确的格式行为可以得到奖励，于是模型开始学习对答案格式的对齐，但是这个过程不是一蹴而就的，如表7所示，在我们的实验中，我们观察到的现象为，在约 50 回合左右，我们的模型开始频繁的出现形如`\box{xxx}` 的答案表示，以不低的频率出现大部分满足我们格式的输出生成（这可能来源于我们奖励函数合理的分级设计）。这种不断的尝试对齐的行为一直持续到 150 回合左右，开始频繁的出现满足格式的答案生成。

<sup>1</sup>需要指出的是，这一现象具有一定的偶然性。我们对比了其他研究者的复现结果，发现他们首次获得正向奖励的步数比我们晚了约 30 到 80 步。



表 7 在完全形成指令遵循前的行为特征

```
<think> Amanda sold 5 friends x 4 tickets each = 20 tickets on the first day. She also sold 32 tickets on the
↪ second day. Together, she sold 20 + 32 = 52 tickets in the first two days. She needs to sell 80 tickets in
↪ total. So, she needs to sell 80 - 52 = 28 tickets on the third day. The answer is \boxed{28}.
```

## 5.2 基于 R1-Zero 的结论

### 5.2.1 强化学习带来的思维强化

我们同样发现了这样的强化学习过程对模型思考时间的线性缩放现象，我们每 80 个强化学习 step 保存一次我们模型的检查点，并针对这些检查点应用 GSM8K 的评测数据集，我们发现我们的模型的推理链路长度随着时间呈现出了线性的关系，这与官方的报告是一致的，在下一小节我们将展开进一步的讨论。

### 5.2.2 数据集评测结果

如图8所示，我们在 GSM8K 的数据集上比较了我们的复现相比于原论文结果的提升，我们的提升是极其显著的，这充分说明了我们的策略的有效性，我们设计的纯 RL 过程显著的提升了模型的推理能力，同时拓展了模型的推理链路，在本报告的后面，我们会进一步探索这种 Scaling 的过程。

表 8 RL 的性能增益

Metric	Accuracy(%)	Format Accuracy(%)	Average Length Per Response
Qwen2.5-1.5B-Instruct	10.08	31.87	133.69
Qwen2.5-1.5B-Instruct-R1-Zero	64.22	94.09	<b>181.95</b>
Qwen2.5-1.5B-Instruct-R1	<b>65.27</b>	<b>97.57</b>	151.84

## 5.3 SFT 的引入与消融

### 5.3.1 SFT 带来的直接增益

如图9所示，我们考虑分析 SFT 后模型的指令遵循能力与答案的准确性。可以看到，微调后，模型在数学上的表现显著提升，同时格式的准确率显著提高<sup>2</sup>，但是可能由于 SFT 引入的 bias，模型的推理链路并没有获得 Scaling 的趋势。

表 9 SFT 的消融验证

Metric	Accuracy(%)	Format Accuracy(%)	Average Length Per Response
Qwen2.5-1.5B-Instruct-SFT	<b>32.68</b>	<b>71.48</b>	113.77
Qwen2.5-1.5B-Instruct	10.08	31.87	<b>133.69</b>

### 5.3.2 SFT 后的 RL

本部分，我们讨论随着强化学习的进行，模型的表现能力的变化，显然，强化学习的进行，模型的准确率和格式正确率都出现了显著的提升，同时 R1 的策略相比于 R1-Zero 表现出了更高的上界与更快的收敛速度，这表明微调的引入是必要的。

## 5.4 RL 引起的思维涌现的报告

我们分析了在不同的强化学习步数下的检查点的测试集推理链路变化，我们关注到，随着强化学习的进行，模型开始逐渐获得本只能在 PRM 中才能获得的推理时间缩放现象，同时其与模型准确性提升的关联也暗示了推理过程的 Scaling 和训练时间的 Scaling 一样都是有效的，这支持了一些近期的工作的观点，在附录B.2我们会给出相应的 Case 进行说明，但是，我们并没有发现自我反思的出现，这很大可能是由于我

<sup>2</sup>由于我们的准确率的计算是建立在格式正确的前提下，所以可能有一定的偏差

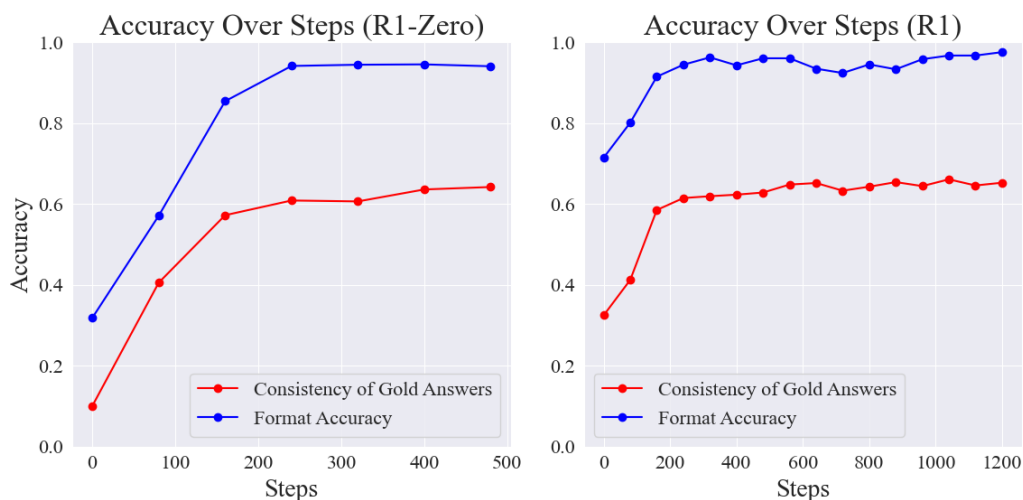


图 1 随着强化学习的进行，模型的答案正确率和格式正确率均出现了提升

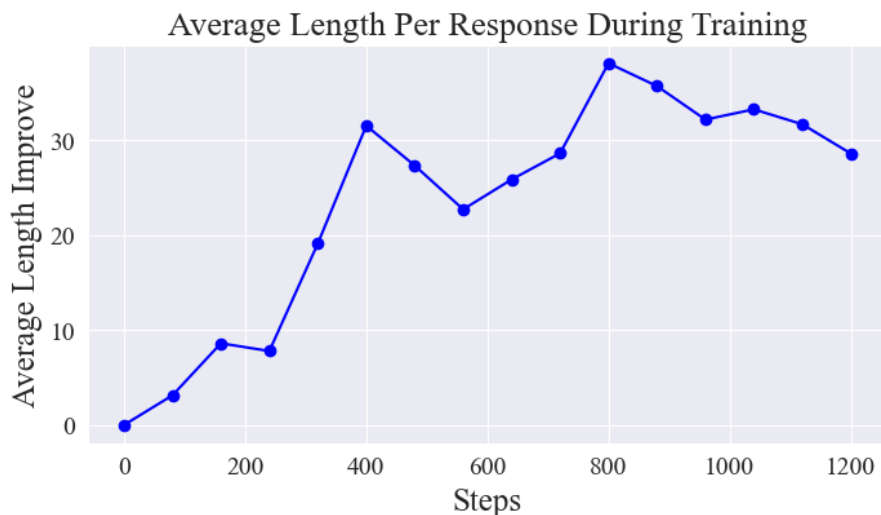


图 2 随着强化学习的进行，模型的推理链路考试出现延长

们的模型太小了，我们也没有发现语种混合的现象，所以在完成了 R1-Zero 的训练后，我们在做 R1 时并没有引入语种的奖励。

## 6 讨论

我们的结果在一定程度上与报告是不一致的，首先很显然的是，我们的提升远远没有对应的蒸馏版本那么显著，即使是我们的 R1 版本，也存在着一定的差距。很明显，这个现象产生的来源仅仅只是我们训练的不充分（缺少了3.4.3节的步骤）以及 SFT 的数据集质量欠佳，但如果仅仅只是作为一个复现的尝试，这样的结果也算是足够充分的了。

但是同样我们发现的问题是，我们的 R1-Zero 版本与报告相比的提升确实显著的，报告指出，对于小模型，R1 的策略并没有那么的显著，但是对蒸馏后的模型进行 RL 或许可以得到不错的效果，而我们的 RL 过程却提升非常显著，这个原因很可能是原论文主要关注的不仅仅是数学领域，而是通用的结果，而我们的复现完全关注于数学，甚至于说就是关注于 GSM8K 的数据集，这可能是我们的效果优于官方的原因。

最后是本实验的强化学习实际上可以和我 1 月份被交叉学科 A 会 WWW 接收的工作<sup>[3]</sup>相结合做一些更有趣的探索，以解决当前 DeepSeek 在实际部署中发现的问题，但是本部分的实验不可能在作业提交前完

成了，同时也涉及到我 NeurIPS 的准备工作，在此就先不做说明了。

## 参考文献

- [1] DEEPSEEK-AI, GUO D, YANG D, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning[A/OL]. 2025. arXiv: 2501.12948. <https://arxiv.org/abs/2501.12948>.
- [2] SHAO Z, WANG P, ZHU Q, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models[A/OL]. 2024. arXiv: 2402.03300. <https://arxiv.org/abs/2402.03300>.
- [3] WEN Z, LIU Z, TIAN Z, et al. Scenario-independent uncertainty estimation for LLM-based question answering via factor analysis[C/OL]//THE WEB CONFERENCE 2025. 2025. <https://openreview.net/forum?id=3UeAN1zicJ>.

## A 冷启动数据集的构建模板

我们使用下面的 prompt 为 GPT-4o 定义了角色：

A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within <think> </think> and <answer> </answer> tags, respectively, i.e., <think> reasoning process here </think><answer> answer here </answer>. Your answer only needs the final number, no additional description is required.

然后我们传入对应的问题，获得答案，然后以此为依据构建附录B.1所示的数据集。

## B Case Study

### B.1 SFT 数据集

我们的 SFT 数据集使用 GPT-4o 构建，每组数据由一对 QA pair 构成，其中 answer 的构建我们使用 xml 的形式分别框出 answer 和 think 部分，我们以 json 列表的形式保存我们的结果。

表 10 SFT 数据集模板

```
[
  {
    "user": "Katrina saw an ad that said if she signed up for her local recycling program, she could earn
    ↪ $5.00. When she signed up, they told her for every friend that she referred, the friend would
    ↪ receive $5.00 and she would receive another $5.00 per friend. That day, she had 5 friends sign
    ↪ up and another 7 friends by the end of the week. How much money in total did she and her friends
    ↪ make?"
  },
  {
    "content": "<think>\nKatrina earns $5.00 for signing up. She also earns $5.00 for each friend she
    ↪ refers. She referred 5 friends on the first day and 7 friends by the end of the week, making it a
    ↪ total of 12 friends. \n\nThe total amount Katrina earns from her friends is 12 friends * $5.00
    ↪ per friend = $60.00. Adding the $5.00 she earned for signing up, Katrina's total earnings are
    ↪ $60.00 + $5.00 = $65.00.\n\nEach friend earns $5.00 for signing up. With 12 friends, the
    ↪ total amount earned by her friends is 12 friends * $5.00 per friend = $60.00.\n\nThe total
    ↪ amount earned by Katrina and her friends is $65.00 + $60.00 =
    ↪ $125.00.\n</think>\n<answer>125</answer>"
  }
]
```

## B.2 推理涌现

我们选取个例子来说明我们的推理涌现：我们选取的问题为：If Lucy would give Linda \$5, Lucy would have the same amount of money as Linda. If Lucy originally had \$20, how much money did Linda have at the beginning?

对应的，我们分别给出，在原始模型下的表现，训练了 240 回合后的表现，R1-Zero 的表现和 R1 的表现，我们刻意选取了表现均较为突出的 case。

- 原始模型:10

- **240 回合 (R1-Zero):**`<think>Let L be the amount of money that Linda originally had. If Lucy gives Linda $5, Lucy will have $20 - $5 = $15. According to the problem, this amount is equal to the amount of money that Linda will have after receiving $5, which is L + $5. Therefore, we have the equation:\n\n15 = L + 5\n\nSubtracting 5 from both sides, we get:\n\n15 - 5 = L\n10 = L\n\nTherefore, Linda originally had $10.</think><answer>10</answer>`

- **R1-Zero:**`<think>Let's denote the amount of money Linda originally had as L. According to the problem, if Lucy gives Linda $5, Lucy would have the same amount of money as Linda. \n\n Lucy originally has $20. After giving $5 to Linda, Lucy would have 20 - 5 = 15 dollars. \n\n At the same time, Linda would have L + 5 dollars after receiving $5 from Lucy. \n\n Since both amounts are equal after this transaction, we can set up the equation:`

$$15 = L + 5$$

\n Solving for L:

$$L = 15 - 5$$

$$L = 10$$

</think><answer>10</answer>

- **R1:**`<think>To solve this problem, I'll start by understanding the relationship between the amounts of money Lucy and Linda have. \n Initially, Lucy has $20, which is given in the problem statement. The key piece of information is that if Lucy gives Linda $5, their money amounts will be equal.\n\n Let's denote the amount of money Linda originally has as L. If Lucy gives Linda $5, her new amount of money becomes 20 - 5 = 15 dollars.\nLinda's new amount of money becomes L + 5 dollars.\n According to the problem, after this exchange, both will have the same amount of money:`

$$20 - 5 = L + 5$$

Simplifying the equation:

$$15 = L + 5$$

Subtracting 5 from both sides gives:

$$L = 10$$

Therefore, Linda originally had \$10.</think><answer>10</answer>