# 1.9.18论文分享

# 论文标题：Re:Form — Reducing Human Priors in Scalable Formal Software Verification with RL in LLMs: A Preliminary Study on Dafny

作者团队：Veri-Code Team*（上海人工智能实验室等）

研究领域：形式化软件验证、大语言模型（LLMs）、强化学习（RL）

## 一，相关名词介绍

### 1.人类先验（Human Priors）

指在模型训练和任务解决过程中，人为注入的知识或约束，包括人工标注的训练数据、人工设计的自然语言思维链（CoT）、依赖人类判断的奖励函数、人工设计的模型结构偏置等。过度依赖人类先验会导致标注成本高、模型泛化能力弱，且可能限制模型自主探索能力。

### 2.监督微调（Supervised Fine-Tuning, SFT）

一种模型训练方法，使用带有标签的数据集对预训练模型进行微调，使模型学习特定任务的映射关系。本文中 SFT 阶段仅使用少量数据，目的是让模型掌握 Dafny 的基本语法和语义，为后续强化学习奠定基础。

### 3.强化学习（Reinforcement Learning, RL）

一种机器学习范式，智能体在环境中通过与环境交互学习最优策略。本文中 RL 以 Dafny 验证器的自动反馈作为奖励信号，引导模型生成更优的形式化规格说明，核心是 Group Relative Policy Optimization（GRPO）算法。

## 二，论文方法

### 1. 过去方法的问题（motivation）

- **验证过程不可靠且扩展性差**：现有基于自然语言的 LLM（如 GPT-4o、Claude）虽能生成代码，但验证过程依赖人工或非形式化方法，无法提供数学上可证明的正确性保证，且难以应对大规模软件验证需求，甚至主流专有模型生成可验证程序的比例极低。
- **过度依赖人类先验**：传统方法需大量人工标注的训练数据（如人工编写形式化规格说明，标注 50 个入门程序需 2 名计算机科学家约 220 小时）、人工设计的自然语言思维链（CoT），以及依赖人类判断的奖励函数。这些人类先验标注成本高、难以规模化，且可能限制模型自主探索能力，导致模型“模仿人类而非真正推理”。
- **数据稀缺与泛化能力弱**：形式化语言的公开数据极少，现有模型在形式化验证任务上数据不足，导致泛化能力差，尤其在多函数组合的复杂任务（如域外测试集）上表现糟糕。
- **评估指标单一**：现有 Dafny 相关基准仅以“验证通过率”作为评估指标，无法区分“表面可验证但语义薄弱的规格说明”和“高质量、强约束的规格说明”，难以准确衡量模型生成规格说明的质量。

## 2. 整体框架

Re:Form 框架分为**数据构建**、**监督微调（SFT）**、**强化学习（RL）** 三个核心阶段，整体流程如下：

### 阶段 1：自动化数据构建（Data Curation）

目的是生成大规模、无人工标注的带形式化规格说明的 Dafny 代码数据集，解决数据稀缺问题，具体分为两条并行 pipeline：

**（1）基于公开 Dafny 资源的 pipeline**

- **步骤 1**：爬取公开 Dafny 代码仓库，例如 GitHub 上的 Dafny 项目，合并公开数据集（如 MetaReflection），筛选出 `.dfy` 文件。
- **步骤 2**：确定性数据清洗：移除重复文件、删除非必要格式（注释、冗余空格、自定义标注）、剔除私有或无关日志语句。
- **步骤 3**：筛选通过 Dafny 验证的代码，最终仅获得约 1.2k 有效样本，反映形式化语言数据稀缺的现状。

**（2）基于 Python 代码转换的 pipeline**（核心，解决数据量不足问题）

- **步骤 1**：规格模板提取：用轻量级解析器分析 Python 函数的函数名、参数类型、返回值、控制结构（循环、条件），映射为 Dafny 规格模板（包含前置条件、后置条件、循环不变式）。
- **步骤 2**：初始转换：将 Python 代码和规格模板作为提示输入 LLM（选择 Claude 3.5

Sonnet，因在 100 个样本测试中转换成功率达 55%，优于 GPT-4o 等模型），生成初始 Dafny 代码。

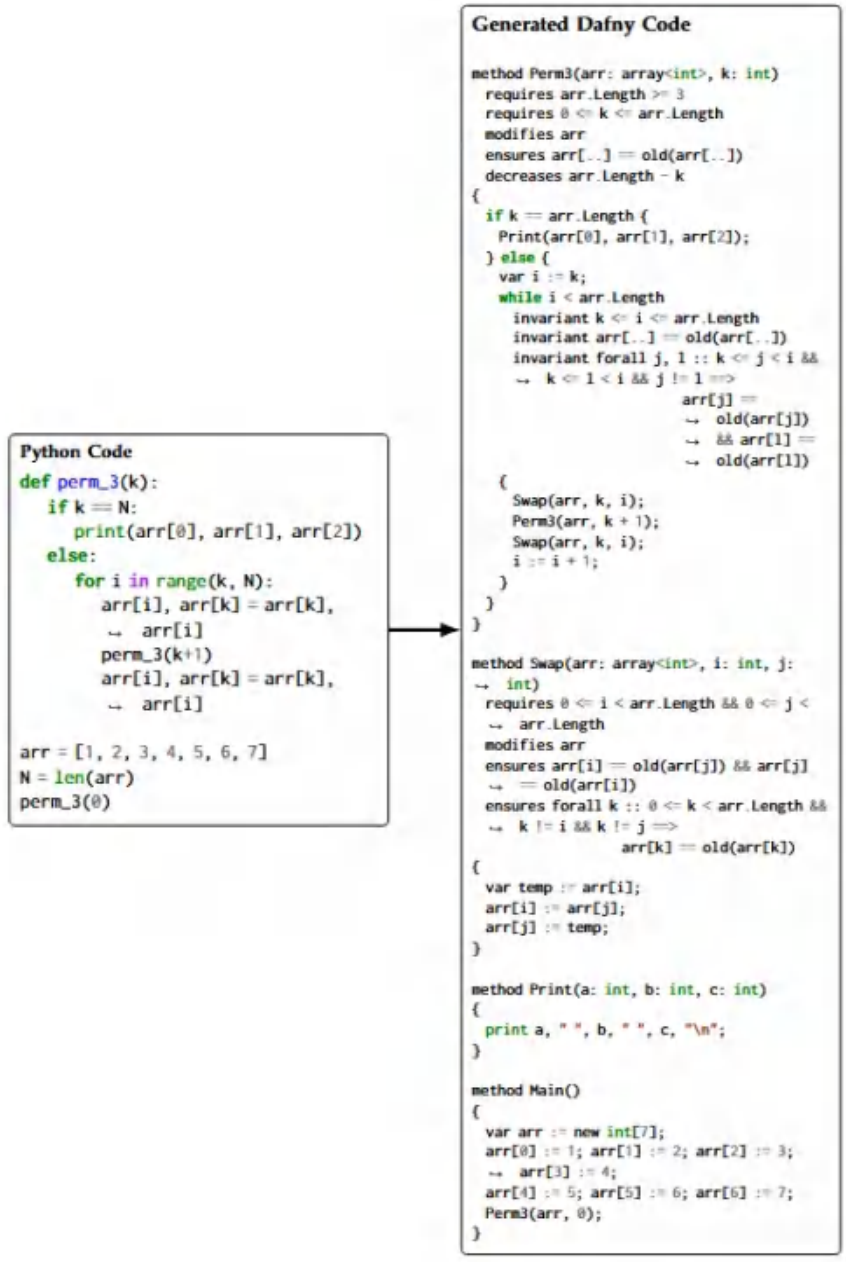
- Claude

| Model | Success ratio (%, out of 100 samples) | Success count |
|---|---|---|
| Claude 3.5 Sonnet | **55.00** | 55 |
| gpt-3.5-turbo | 45.00 | 45 |
| gpt-4o | 31.00 | 31 |
| gpt-4o-mini | 41.00 | 41 |
| o1 | 36.00 | 36 |
| o1-mini | 33.00 | 33 |
| o3-mini | 37.00 | 37 |
| gemini-2.0-flash | 38.00 | 38 |

**Table C.1** Model Conversion Success Rate Comparison

  - **步骤 3**：自动验证与调试（迭代最多 10 次）：将初始 Dafny 代码输入 Dafny 验证器，若验证失败，收集错误信息并生成调试提示，让 LLM 修正代码，直至验证通过或达到迭代上限。
  - **步骤 4**：规格插入与二次验证：将生成的规格说明插入 Dafny 代码的主函数和子函数，再次执行验证与调试流程，最终保存通过验证的代码。该 pipeline 无需人工逐样本标注，仅需设计初始模板和抽查质量，生成 20k 有效 Dafny 样本。

▼ 示例




### 阶段 2：监督微调（SFT）

目的是让模型掌握 Dafny 的基本语法和语义，为 RL 阶段奠定基础，具体步骤：

**（1）数据选择**：从阶段 1 生成的 20k 数据集中选择 3k 样本作为 SFT 训练集，确保数据中无自然语言 CoT 和代码注释，减少人类先验。

**（2）任务定义**：将任务形式化为“输入代码实现 `c`，输出带规格说明的完整代码 `y`”，即 `c ⊕ y = π(c)`，选择“生成完整程序”而非“规格填充”，避免代码插入位置的额外挑战。

**（3）超参数优化**：基于 Qwen-2.5 架构的模型（0.5B~14B 参数），使用 Deepspeed ZeRO Stage 3 优化，采用余弦学习率调度（10% 预热），通过网格搜索确定各模型大小的最优超参数（如 0.5B 模型的梯度累积步长为 4、学习率为 0.75e-4、训练轮次为 10）。

**（4）训练约束**：控制训练数据量和计算预算，避免模型过拟合，最终小模型（如 0.5B）也能生成语法正确的 Dafny 代码。

### 阶段 3：强化学习（RL）探索

目的是让模型在无额外人类指导的情况下，通过与 Dafny 验证器交互，自主提升生成规格说明的质量和泛化能力，具体步骤：

**(1)环境与反馈机制**：以 Dafny 验证器为环境，验证器的输出（语法是否正确、规格是否可验证、规格是否优于真值）作为自动反馈信号，无需人工判断，确保 scalability。

**(2)奖励设计**（三类奖励，解决“仅验证通过但规格薄弱”的问题）

- **语法奖励（Syntax Reward）**：若生成的 Dafny 代码通过编译（无语法错误、类型错误），则给予奖励，作为低成本的正确性代理信号。
- **验证奖励（Verification Reward）**：若生成的规格说明与代码通过 Dafny 验证（即规格说明与代码逻辑一致），则给予奖励，遵循现有 Dafny 基准的评估逻辑。
- **子集奖励（Subset Reward）**：若生成的规格说明“优于”真值（前置条件更弱 + 后置条件更强），则给予奖励，通过 Dafny 验证器证明两个逻辑蕴含关系实现：
  - 前置条件松弛：`GT_pre ⇒ GEN_pre`（生成的前置条件包含真值前置条件的所有合法输入）；
  - 后置条件强化：`GT_pre ⇒ (GEN_post ⇒ GT_post)`（在真值前置条件下，生成的后置条件满足时，真值后置条件必满足）。

**(3)RL 算法与正则化**

- 算法选择：采用 Group Relative Policy Optimization（GRPO），目标函数为：

and compute the objective $\mathcal{J}_{\text{GRPO}}(\theta)$, which is

$$\mathbb{E}_{\substack{c\sim P(C) \\ \{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|c)}}\left[\frac{1}{G}\sum_{i=1}^{G}\min\left(\frac{\pi_\theta(y_i\mid c)}{\pi_{\theta_{\text{old}}}(y_i\mid c)}A_i,\ \text{clip}\left(\frac{\pi_\theta(y_i|c)}{\pi_{\theta_{\text{old}}}(y_i|c)},1-\epsilon,1+\epsilon\right)A_i\right)-\beta\,D_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}})\right], \tag{1}$$

where $\pi_\theta$ and $\pi_{\theta_{\text{old}}}$ are the current policy model and data generation model and $A_i$ is the group-wise advantage:

$$A_i=\frac{r_i-\text{mean}(\{r_j\}_{j=1}^{G})}{\text{std}(\{r_j\}_{j=1}^{G})}. \tag{2}$$

- *组内相对优势Ai*：若 *yi* 的奖励远高于同组平均，*Ai* 为正且绝对值大，说明它是“组内最优候

选”；

- min和 `clip(...)` ：进一步限制更新幅度，保证训练稳定；
- 另外，引入了“熵奖励”，让模型生成的内容更多样，不会总是生成重复或者模式单一的结果。最终是用“子集奖励 + KL 散度 + 熵奖励”这样的组合来平衡模型探索新内容和保持稳定的能力。

**(4)RL 训练设置**

每个输入生成 4 个样本（采样温度 1.0），批量大小 1024，学习率 1e−5，KL 系数 0.01，熵系数 0.02；使用 A800−SXM4−80G GPU，3B 模型训练 40 轮约需 20 小时（4 节点 ×8 GPU）。

## 3.核心难点解析

**（1）如何设计自动化数据构建 pipeline 以减少人工标注?**

- **难点**：形式化语言数据稀缺，人工标注成本极高（标注 50 个程序需 220 小时），无法满足大规模训练需求。
- **解决方案**：设计两条无人工标注的 pipeline：
  - 对公开 Dafny 资源进行清洗，但仅获 1.2k 样本；
  - 以 Python 代码为输入，通过“规格模板提取→LLM 初始转换→迭代验证调试”流程，自动生成带规格说明的 Dafny 代码，生成 20k 样本。关键在于利用 LLM 生成初始代码，再通过 Dafny 验证器的错误反馈引导 LLM 迭代修正，无需人工逐样本干预，仅需设计模板和抽查质量。

**（2）如何避免强化学习中模型“奖励黑客”（生成薄弱规格以通过验证）？**

- **难点**：若仅使用“验证奖励”，模型会生成语义薄弱但易验证的规格说明（如 `ensures -1.111 == -1.111` 这类恒真语句），虽能通过验证，但无法准确描述程序行为，导致评估指标失真。
- **解决方案**：引入“子集奖励”，通过逻辑蕴含关系判断生成规格是否“优于”真值（前置条件更弱 + 后置条件更强）。该奖励迫使模型不仅要通过验证，还要生成约束更严格、信息更丰富的规格说明，例如在“寻找公主位置”的任务中，模型需额外覆盖“无公主时返回 (−1,−1)”的场景，而非仅满足真值的基本约束。

**（3）如何平衡强化学习的探索与稳定，避免模式崩溃?**

- **难点**：RL 阶段模型易出现“模式崩溃”（仅生成少数几种可验证的规格，缺乏多样性），或因

探索过度导致性能不稳定（如生成大量语法错误代码）。

- **解决方案**：
  - 算法层面：采用 GRPO 算法，通过群体采样计算相对优势，避免单一样本的极端奖励影响策略更新；
  - 正则化层面：引入 KL 散度正则化（锚定 SFT 模型的基础能力，防止策略偏离过大）和熵奖励（增加生成多样性，鼓励探索新规格）；
  - 实验验证：仅用验证奖励或无正则化时，模型多样性低且 SSR 低；加入“KL + 熵”正则化后，模型在 pass@128（128 次采样中至少一次通过的比例）上显著提升，且生成规格的嵌入空间方差更高（多样性更强）。

**（4）如何评估模型生成规格说明的质量，而非仅关注验证通过率？**

- **难点**：现有基准仅用“验证通过率”作为指标，无法区分“高质量规格”和“薄弱规格”，导致模型优化方向偏离任务核心（准确描述程序行为）。
- **解决方案**：
  - 提出“规格优越性率（SSR）”：通过 Dafny 验证器证明生成规格与真值的逻辑关系，衡量规格的强度和准确性；
  - 引入“新颖规格率（Novel Spec Rate）”：判断生成规格是否未出现在 SFT 模型的 128 次采样中，且语义上更强（非简单重述或恒真语句），衡量模型的探索能力；
  - 实验验证：设计 DafnyComp 基准：包含多函数组合的域外测试集，评估模型的组合推理和泛化能力，而非仅在简单单函数任务上表现优异。

# 三、实验结果与分析

## 1.实验设置

**（1）模型选择：**

基于 Qwen-2.5 架构的模型，参数规模 0.5B~14B（0.5B 模型通过知识蒸馏获得，从 7B/14B SFT 模型蒸馏，采用 JSD 或 RKL 损失，温度 1~2）；对比模型包括 GPT-4o、Claude 3.5 Sonnet、Qwen 预训练基础模型。

**（2）数据集设置**

- 训练集：SFT 用 3k 样本，RL 用 4.5k 样本（均来自阶段 1 的 Python 转换 pipeline）；
- 评估集：
  - 域内：512 个未见过的 Python 转换 Dafny 样本；
  - 域外：300 个 DafnyComp 基准的多函数组合样本；
  - 对比基准：100 个 DafnyBench 样本（现有最大 Dafny 基准）。

**（3）评估指标**

- 验证率（Validation Rate）：生成代码无语法错误的比例；
- 验证通过率（Verification Rate）：生成规格说明通过 Dafny 验证的比例；
- 规格优越性率（SSR）：生成规格优于或等价于真值的比例；
- pass@K：K 次采样中至少一次满足指标（如验证通过、SSR 达标）的比例（K=1,128）。

**（4）对比方法**

- 预训练基础模型（Qwen-Coder-0.5B~14B）：无任何微调，作为 baseline；
- 专有模型（GPT-4o、Claude 3.5 Sonnet）：零样本生成，作为行业标杆；
- SFT 模型（仅 SFT 无 RL）：验证 SFT 阶段的基础性能；
- RL 模型（不同奖励配置）：验证不同奖励和正则化的效果，包括“仅验证奖励”“子集奖励”“子集奖励 + KL + 熵”。

## 2.实验结果

**（1）SFT 阶段：小模型已超越专有模型**

### C.2.1 SFT Results

| Model | Validation Rate (%) | Verificaion Rate (%) | Spec Superiority Rate (%) |
|---|---|---|---|
| GPT-4o | 47.7 | 12.1 | 7.0 |
| Qwen-Coder-0.5B | 3.5 | 1.6 | 0.0 |
| Qwen-Coder-1.5B | 5.5 | 1.2 | 0.0 |
| Qwen-Coder-3B | 6.6 | 2.3 | 0.2 |
| Qwen-Coder-7B | 17.6 | 3.7 | 0.0 |
| Qwen-Coder-14B | 5.9 | 2.5 | 0.4 |
| 0.5B SFT | 80.1 | 33.6 | 18.0 |
| 1.5B SFT | 84.2 | 41.6 | 22.1 |
| 3B SFT | 88.7 | 48.0 | 26.6 |
| 7B SFT | 90.8 | 53.3 | 27.9 |
| 14B SFT | 94.3 | 62.9 | 34.2 |

**Table C.2** Our SFT models already show a significant improvement from the base model and surpass the powerful model, GPT-4o.

**（2）RL 阶段：进一步提升质量与泛化**

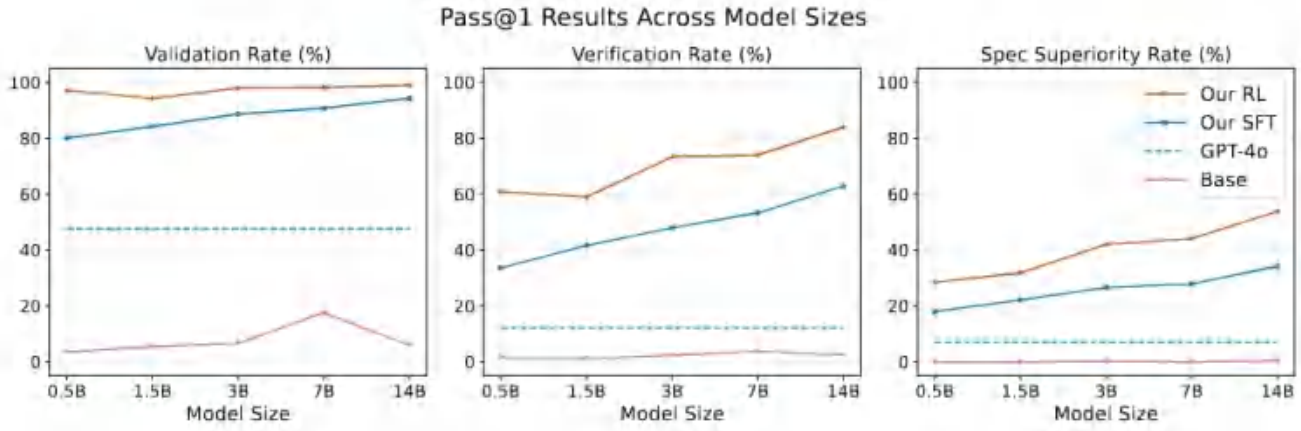



**Figure 3.2** The figure shows the comparison between GPT-4o, our Qwen base models, SFT models and RL-trained models scaling over model size on our in-domain evaluation set. The pass@1 improvement of SFT and subsequent RL over our base models is substantial.

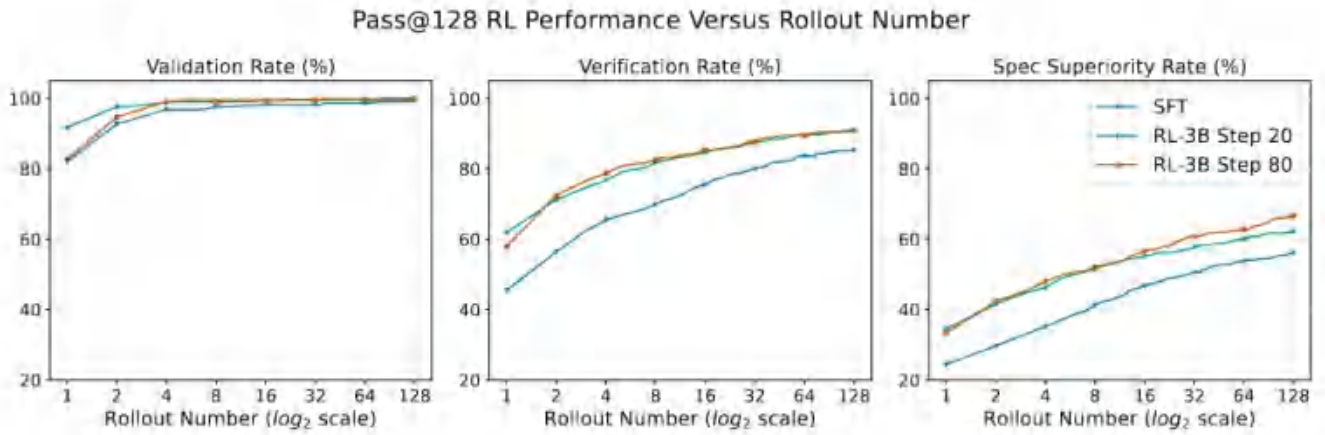



**Figure 3.3** The figure reports SFT and RL performance with 128 rollouts. The plotted rate measures whether at least one rollout attains the corresponding reward. RL yields a clear improvement from SFT, indicating genuine quality gains rather than mere compression of rollouts.

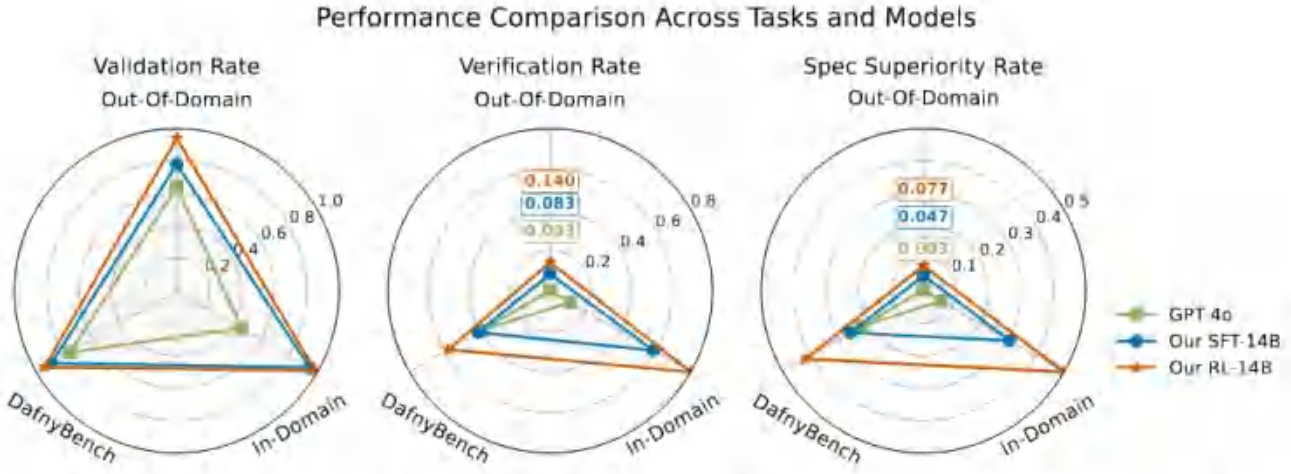



**Figure 3.5** Our 14B RL model dominates the pass@1 performance over SFT and GPT-4o, the best performing proprietary LLM other than our data-generator, Claude. Notably, GPT-4o attains the best score on DafnyBench, highlighting an asymmetry toward that benchmark.

### (3) 奖励与正则化的 ablation 实验

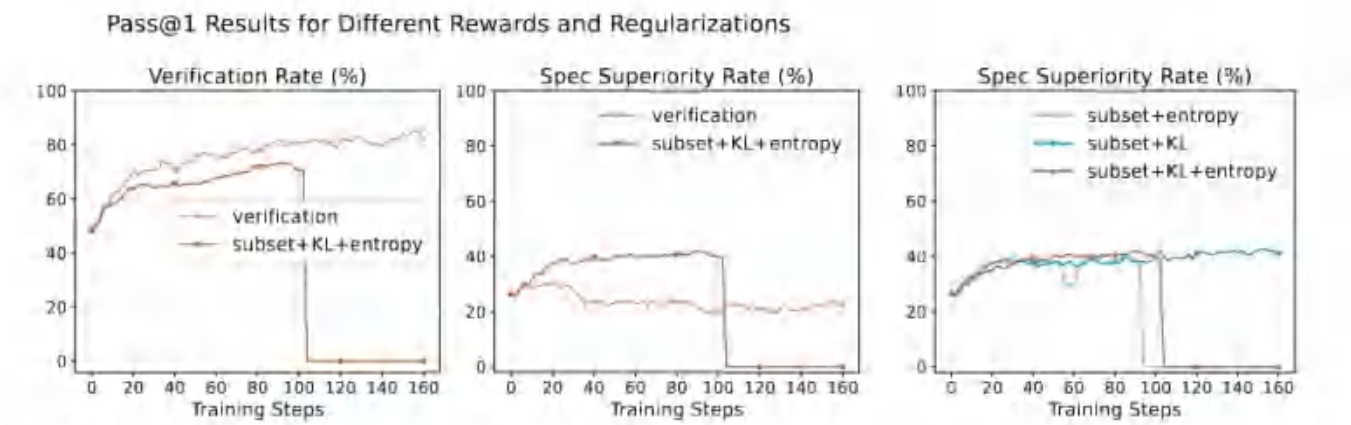



**Figure 3.6** These figures present the training curves for different reward schemes and regularization choices. The left figure shows that using the subset reward stops the quality drop, demonstrated by the spec superiority rate. The right figure shows that entropy regularization leads to instability in training, and all regularization choices show similar pass@1 performance before crashing.

### (4) 模型规模的影响

- 性能随规模增长：从 0.5B 到 14B，模型的验证率、验证通过率、SSR 均稳步提升，如 SSR 从 30.1%（0.5B RL）提升至 53.9%（14B RL），证明更大模型有更强的形式化推理能力。
- 效率优势：0.5B RL 模型的 SSR 达 30.1%，超过 GPT-4o（7.0%）和基础模型（0%），且训练仅需 16 个 GPU、11 小时，相比 14B 模型（64 个 GPU、36 小时）成本更低，证明小模型在特定形式化任务上也能有效工作。

### (5) 定性分析：模型生成新颖规格

**Code**

```
method findPrincess(n: int, grid: seq<seq<char>>) returns (position:
↪  (int, int))
{
 var i := 0;
 position := (-1, -1);
 while i < n
 {
  var j := 0;
  while j < n
  {
   if grid[i][j] == 'p' {
    position := (i, j);
    return;
   }
   j := j + 1;
  }
  i := i + 1;
 }
}
```

**Ground Truth**

```
requires n > 0
requires |grid| == n
requires forall i :: 0 <= i < n ==> |grid[i]| == n
requires exists i, j :: 0 <= i < n && 0 <= j < n && grid[i][j] == 'p'
ensures 0 <= position.0 < n && 0 <= position.1 < n
ensures grid[position.0][position.1] == 'p'
```

**Verification Reward Generation**

```
requires n > 0
requires |grid| == n && forall i :: 0 <= i < n ==> |grid[i]| == n
ensures -1 <= position.0 < n && -1 <= position.1 < n
```

**Subset Reward Generation**

```
requires n >= 0
requires |grid| == n && n >= 0
ensures position.0 == -1 && position.1 == -1 ==>
   forall i, j :: 0 <= i < n && 0 <= j < n ==> grid[i][j] != 'p'
ensures position.0 != -1 && position.1 != -1 ==>
   0 <= position.0 < n && 0 <= position.1 < n &&
   grid[position.0][position.1] == 'p'
```

**Figure C.3** The top block shows the input code, followed by the extracted preconditions and postconditions for three cases: the ground-truth specification, the output from the model trained with verification reward, and the output from the model trained with subset reward. The subset reward model produces a strictly stronger specification, capturing an additional behavior (the case with no princess in the grid) that is not covered by the ground-truth, thus demonstrating superior logical coverage.

# 四、结论

## 1. 论文的贡献

**（1）提出减少人类先验的形式化软件验证框架**：通过自动化数据构建（无人工标注）、移除自然语言 CoT、基于验证器反馈的自动奖励，大幅减少对人类先验的依赖，解决了传统方法标注成本高、扩展性差的问题。

**（2）设计高质量的奖励与评估体系**：引入“子集奖励”解决“薄弱规格”问题，提出“SSR”“新颖规格率”等指标，结合 DafnyComp 基准，实现对规格说明质量、探索能力、泛化能力的全面评估，弥补了现有基准指标单一的缺陷。

**（3）验证小模型的有效性**：0.5B 参数的 SFT 模型已能超越 GPT-4o 等专有模型，RL 进一步提升性能，证明在特定形式化任务中，小模型通过合理训练策略可实现高效推理，降低计算成本。

**（4）开源资源推动领域发展**：开源了完整的 pipeline（代码、数据、模型 checkpoint，https://github.com/Veri-Code/ReForm，https://huggingface.co/Veri-Code），为形式化软件验证与 LLM 结合的研究提供基础数据和工具。

## 2. 论文的限制

**（1）任务范围局限**：当前任务聚焦于“从代码实现生成形式化规格说明”，未覆盖更复杂的形式化验证场景（如自动定理证明、多语言形式化转换）；且实验任务以算法、数据结构为主，未涉及工业级大型软件的验证，泛化到复杂真实场景仍需验证。

**（2）奖励函数的潜在偏置**：虽减少人类先验，但奖励函数的设计（如子集奖励的逻辑蕴含规则、KL / 熵系数的选择）仍依赖人工经验，可能引入隐性人类偏置；且“新颖规格”的判断依赖 SFT 样本和真值，若真值本身存在缺陷，可能影响评估准确性。

**（3）训练稳定性问题**：强化学习阶段，加入熵正则化后模型训练易出现不稳定（如训练后期 SSR 突然下降），且部分模型在生成完整代码后会额外输出无意义 tokens（占比 80%），虽不影响验证，但浪费计算资源，需进一步优化训练策略。

**（4）验证器的局限性依赖**：框架完全依赖 Dafny 验证器的反馈，若验证器存在“不完全性”（即无法证明某些合法规格），会导致模型错过优质规格；且 Dafny 语言的应用范围有限，无法直接迁移到 C、Java 等主流语言的形式化验证（如 VeriFast 工具）。

## 3. 未来的方向

**（1）扩展任务与语言范围**：将框架扩展到“自动定理证明”“形式化规格修复”“多语言形式化转换（如 C→Dafny）”等任务；适配 VeriFast（C/Java）、Lean 4（数学推理）等工具，扩大应用场景。

**（2）优化奖励与训练稳定性**：探索自适应奖励函数（如基于验证器错误类型动态调整奖励权重），解决熵正则化导致的训练不稳定问题；设计更高效的 RL 算法，减少无意义 token 生成，提升训练效率。

**（3）工业级软件验证适配**：针对工业级软件的复杂性（如多模块、并发、实时系统），优化数据构建 pipeline（如从工业代码中提取规格模板），设计支持增量验证的 RL 策略，推动框架在安全关键领域（如自动驾驶、医疗设备）的实际应用。