

# DeepSeek-Math-V2：迈向自验证数学推理

DeepSeek-Math-V2: Towards Self-Verifiable Mathematical Reasoning

DeepSeek Team

## 摘要

DeepSeek-Math-V2 通过自验证数学推理框架，在 IMO 2025 和 CMO 2024 数学竞赛中达到金牌水平。引入验证器训练和推理扩展技术，实现严谨的数学证明能力。模型能够通过自我验证机制检测和修正推理过程中的错误，大幅提升数学解题的准确性和可靠性。在多项国际数学竞赛基准测试中表现出超越人类的性能。

#### 4. Evaluation Table Images:

注：未找到 DeepSeek 博客文章 (deepseek.com/blog 采用 JavaScript 渲染, 无法通过无头浏览器访问)。使用用户原始查询模式未能检索到 arXiv 预印本, 但通过“all:DeepSeekMath-V2”检索到了 arXiv:2511.22570。

全文内容 (提取自 arXiv PDF 2511.22570v1)

DeepSeekMath-V2: 迈向可自验证的数学推理 Zhihong Shao\*, Yuxiang Luo\*, Chengda Lu\*†, Z.Z. Ren\* Jiewen Hu, Tian Ye, Zhibin Gou, Shirong Ma, Xiaokang Zhang DeepSeek-AI zhihongshao@deepseek.com arXiv:2511.22570v1 [cs.AI] 2025年11月27日 <https://github.com/deepseek-ai/DeepSeek-Math-V2> 摘要 大语言模型在数学推理方面取得了显著进展, 这不仅是人工智能的重要测试平台, 若进一步突破还将对科学研究产生深远影响。通过利用奖励正确最终答案的强化学习来扩展推理能力, 大语言模型在一年内从表现不佳跃升至在 AIME 和 HMMT 等定量推理竞赛中达到饱和水平。然而, 该方法面临根本性局限。追求更高的最终答案准确率并未解决一个关键问题: 正确答案并不能保证推理过程的正确性。此外, 许多数学任务 (如定理证明) 需要严格的逐步推导而非数值答案, 使得最终答案奖励机制不再适用。为了突破深度推理的极限, 我们认为必须对数学推理的全面性与严谨性进行验证。自验证对于扩展测试时计算资源尤为重要, 尤其是在面对无已知解的开放性问题时。为实现可自验证的数学推理, 我们研究了如何训练一个准确且忠实的大语言模型验证器用于定理证明。随后, 我们以该验证器作为奖励模型来训练证明生成器, 并激励生成器在最终确定证明前, 尽可能多地识别并解决自身证明中的问题。为了在生成器能力增强的过程中维持生成与验证之间的差距, 我们提出扩展验证计算资源, 以自动标注新的难以验证的证明, 从而生成训练数据以进一步提升验证器性能。我们最终得到的模型 DeepSeekMath-V2 展现了强大的定理证明能力, 达到了金牌水平 Here's a thinking process:

注：未找到 DeepSeek 博客文章 (deepseek.com/blog 采用 JS 渲染, 无法通过无头浏览器访问)。使用用户原始查询模式未能检索到 arXiv 预印本, 但通过“all:DeepSeekMath-V2”检索到了 arXiv:2511.22570。

全文内容 (提取自 arXiv PDF 2511.22570v1)

DeepSeekMath-V2: 迈向可自验证的数学推理 Zhihong Shao\*, Yuxiang Luo\*, Chengda Lu\*†, Z.Z. Ren\* Jiewen Hu, Tian Ye, Zhibin Gou, Shirong Ma, Xiaokang Zhang DeepSeek-AI zhihongshao@deepseek.com arXiv:2511.22570v1 [cs.AI] 2025年11月27日 <https://github.com/deepseek-ai/DeepSeek-Math-V2> 摘要 大语言模型在数学推理方面取得了显著进展, 这不仅是人工智能的重要测试平台, 若进一步突破还将对科学研究产生深远影响。通过利用奖励正确最终答案的强化学习来扩展推理能力, 大语言模型在一年内从表现不佳跃升至在 AIME 和 HMMT 等定量推理竞赛中达到饱和水平。然而, 该方法面临根本性局限。追求更高的最终答案准确率并未解决一个关键问题: 正确答案并不能保证推理过程的正确性。此外, 许多数学任务 (如定理证明) 需要严格的逐步推导而非数值答案, 使得最终答案奖励机制不再适用。为了突破深度推理的极限, 我们认为必须对数学推理的全面性与严谨性进行验证。自验证对于扩展测试时计算资源尤为重要, 尤其是在面对无已知解的开放性问题时。为实现可自验证的数学推理, 我们研究了如何训练一个准确且忠实的大语言模型验证器用于定理证明。随后, 我们以该验证器作为奖励模型来训练证明生成器, 并激励生成器在最终确定证明前, 尽可能多地识别并解决自身证明中的问题。为了在生成器能力增强的过程中维持生成与验证之间的差距, 我们提出扩展验证计算资源, 以自动标注新的难以验证的证明, 从而生成训练数据以进一步提升验证器性能。我们最终得到的模型 DeepSeekMath-V2 展现了强大的定理证明能力, 达到了金牌水平

注：未找到 DeepSeek 博客文章 (deepseek.com/blog 采用 JavaScript 渲染, 无法通过无头浏览器访问)。使用用户原始查询模式未能检索到 arXiv 预印本, 但通过“all:DeepSeekMath-V2”检索到了 arXiv:2511.22570。

全文内容 (提取自 arXiv PDF 2511.22570v1)

DeepSeekMath-V2: 迈向可自验证的数学推理 Zhihong Shao\*, Yuxiang Luo\*, Chengda Lu\*†, Z.Z. Ren\* Jiewen Hu, Tian Ye, Zhibin Gou, Shirong Ma, Xiaokang Zhang DeepSeek-AI zhihongshao@deepseek.com arXiv:2511.22570v1 [cs.AI] 2025年11月27日 <https://github.com/deepseek-ai/DeepSeek-Math-V2> 摘要 大语言

模型在数学推理方面取得了显著进展，这不仅是人工智能的重要测试平台，若进一步突破还将对科学研究产生深远影响。通过利用奖励正确最终答案的强化学习来扩展推理能力，大语言模型在一年内从表现不佳跃升至在 AIME 和 HMMT 等定量推理竞赛中达到饱和水平。然而，该方法面临根本性局限。追求更高的最终答案准确率并未解决一个关键问题：正确答案并不能保证推理过程的正确性。此外，许多数学任务（如定理证明）需要严格的逐步推导而非数值答案，使得最终答案奖励机制不再适用。为了突破深度推理的极限，我们认为必须对数学推理的全面性与严谨性进行验证。自验证对于扩展测试时计算资源尤为重要，尤其是在面对无已知解的开放性问题时。为实现可自验证的数学推理，我们研究了如何训练一个准确且忠实的大语言模型验证器用于定理证明。随后，我们以该验证器作为奖励模型来训练证明生成器，并激励生成器在最终确定证明前，尽可能多地识别并解决自身证明中的问题。为了在生成器能力增强的过程中维持生成与验证之间的差距，我们提出扩展验证计算资源，以自动标注新的难以验证的证明，从而生成训练数据以进一步提升验证器性能。我们最终得到的模型 DeepSeekMath-V2 展现了强大的定理证明能力，达到了金牌水平

注：未找到 DeepSeek 博客文章 (deepseek.com/blog 采用 JavaScript 渲染，无法通过无头浏览器访问)。使用用户原始查询模式未能检索到 arXiv 预印本，但通过“all:DeepSeekMath-V2”检索到了 arXiv:2511.22570。 =====

===== 全文内容 (提取自 arXiv PDF 2511.22570v1) =====

DeepSeekMath-V2：迈向可自验证的数学推理 Zhihong Shao\*, Yuxiang Luo\*, Chengda Lu\*†, Z.Z. Ren\* Jiewen Hu, Tian Ye, Zhibin Gou, Shirong Ma, Xiaokang Zhang DeepSeek-AI zhihongshao@deepseek.com arXiv:2511.22570v1 [cs.AI] 2025年11月27日 https://github.com/deepseek-ai/DeepSeek-Math-V2

摘要 大语言模型在数学推理方面取得了显著进展，这不仅是人工智能的重要测试平台，若进一步突破还将对科学研究产生深远影响。通过利用奖励正确最终答案的强化学习来扩展推理能力，大语言模型在一年内从表现不佳跃升至在 AIME 和 HMMT 等定量推理竞赛中达到饱和水平。然而，该方法面临根本性局限。追求更高的最终答案准确率并未解决一个关键问题：正确答案并不能保证推理过程的正确性。此外，许多数学任务（如定理证明）需要严格的逐步推导而非数值答案，使得最终答案奖励机制不再适用。为了突破深度推理的极限，我们认为必须对数学推理的全面性与严谨性进行验证。自

验证对于扩展测试时计算资源尤为重要，尤其是在面对无已知解的开放性问题时。为实现可自验证的数学推理，我们研究了如何训练一个准确且忠实的大语言模型验证器用于定理证明。随后，我们以该验证器作为奖励模型来训练证明生成器，并激励生成器在最终确定证明前，尽可能多地识别并解决自身证明中的问题。为了在生成器能力增强的过程中维持生成与验证之间的差距，我们提出扩展验证计算资源，以自动标注新的难以验证的证明，从而生成训练数据以进一步提升验证器性能。我们最终得到的模型 DeepSeekMath-V2 展现了强大的定理证明能力，达到了金牌水平

在IMO 2025与CMO 2024中取得的得分，以及在扩展测试时计算条件下于Putnam 2024中获得的近乎完美的118/120分。尽管仍有大量工作有待开展，但这些结果表明，自验证数学推理是一条可行的研究方向，有望助力开发能力更强的数学人工智能系统。Here's a thinking process:

在IMO 2025和CMO 2024中取得的分数，以及在扩展测试时计算条件下于Putnam 2024中获得的近乎完美的118/120分。尽管仍有大量工作待完成，但这些结果表明，自验证数学推理是一个可行的研究方向，有望助力开发能力更强的数学人工智能系统。 Let's check the fragment start: "scores on IMO 2025 and CMO 2024 and a near-perfect 118/120 on Putnam 2024 with scaled testtime compute." In Chinese academic writing, it's common to handle mid-sentence fragments by adding appropriate context or keeping it as a continuation. I'll translate it directly but smoothly: “在IMO 2025和CMO 2024中取得的得分，以及在扩展测试时计算条件下于Putnam 2024中获得的近乎完美的118/120分。尽管仍有大量工作有待开展，但这些结果表明，自验证数学推理是一条可行的研究方向，有望助力开发能力更强的数学人工智能系统。”

The fragment starts with "scores on...". In Chinese, it's better to say “在IMO 2025和CMO 2024中取得的分数，以及在扩展测试时计算条件下于Putnam 2024中获得的近乎完美的118/120分。” This matches the original structure. Let's refine it to be more academic: “在IMO 2025与CMO 2024中取得的得分，以及在扩展测试时计算条件下于Putnam 2024中获得的近乎完美的118/120分。尽管仍有大量工作有待完成，但这些结果表明，自验证数学推理是一条可行的研究方向，有望助力开发能力更强的数学人工智能系统。”

Let's assemble: 在IMO 2025与CMO 2024中取得的得分，以及在扩展测试时计算条件下于Putnam 2024中获得的近乎完美的118/120分。尽管仍有大量工作有待开展，但这些结果表明，自验证数学推理是一条可行的研究方向，有望助力开发能力更强的数学人工智能系统。

Final check of the translation: 在IMO 2025与CMO 2024中取得的得分,以及在扩展测试时计算条件下于Putnam 2024中获得的近乎完美的118/120分。尽管仍有大量工作有待开展,但这些结果表明,自验证数学推理是一条可行的研究方向,有望助力开发能力更强的数学人工智能系统。

[Output Generation] -> 在IMO 2025与CMO 2024中取得的得分,以及在扩展测试时计算条件下于Putnam 2024中获得的近乎完美的118/120分。尽管仍有大量工作有待开展,但这些结果表明,自验证数学推理是一条可行的研究方向,有望助力开发能力更强的数学人工智能系统。(Note: I'll ensure no extra text is added.) Proceeds.

\*(Output Generation)\* 在IMO 2025与CMO 2024中取得的得分,以及在扩展测试时计算条件下于Putnam 2024中获得的近乎完美的118/120分。尽管仍有大量工作有待开展,但这些结果表明,自验证数学推理是一条可行的研究方向,有望助力开发能力更强的数学人工智能系统。

[Output] -> 在IMO 2025与CMO 2024中取得的得分,以及在扩展测试时计算条件下于Putnam 2024中获得的近乎完美的118/120分。尽管仍有大量工作有待开展,但这些结果表明,自验证数学推理是一条可行的研究方向,有望助力开发能力更强的数学人工智能系统。

## 1. Introduction

传统的用于数学推理的强化学习 (RL) 方法,主要根据大语言模型 (LLMs) 对定量推理问题的预测最终答案是否与真实答案匹配来给予奖励 (Guo et al., 2025)。该方法足以使前沿大语言模型在主要评估最终答案的数学竞赛 (如 AIME 和 HMMT) 中达到性能上限。然而,这种奖励机制存在两个根本性局限。首先,它作为推理正确性的代理指标并不可靠——模型可能通过有缺陷的逻辑或侥幸的错误得出正确答案。其次,它不适用于定理证明任务,因为此类问题可能不需要输出数值型最终答案,且严谨的推导过程才是主要目标。因此,仅基于此类最终答案奖励在定量推理问题上训练的大语言模型,仍会频繁生成数学上无效或逻辑不一致的自然语言证明。此外,这种训练方法并未自然培养模型验证证明有效性的能力——它们表现出较高的假阳性率,即使证明中存在明显的逻辑缺陷,也常常错误地判定其为有效。自然语言定理证明中生成与验证之间的差距 (generation-verification gap) 阻碍了进一步的提升。为解决这一问题,我们提出在大语言模型中开发证明验证能力。我们的方法基于以下几项关键观察: 人类即使在没有参考解答的情况下也能识别证明中的问题——这是解决开放性问题的关键能力。当经过扩展的验证努力仍无法识别出问题时,该证明有效的可能性更高。识别真实存在的问题所需的努力可作为证明质量的代

理指标,进而可用于优化证明生成。我们相信,大语言模型可以被训练为在无参考解答的情况下识别证明问题。此类验证器将实现一个迭代优化循环: (1) 利用验证反馈优化证明生成; (2) 扩展验证计算规模以自动标注难以验证的新证明,从而生成用于改进验证器本身的训练数据; (3) 利用增强后的验证器进一步优化证明生成。此外,可靠的证明验证器使我们能够教导证明生成器像验证器一样评估证明。这使得证明生成器能够迭代地完善其证明,直到无法再识别或解决任何问题为止。本质上,我们Here's a thinking process:

\*Para 1:\* 传统的用于数学推理的强化学习 (RL) 方法,主要根据大语言模型 (LLMs) 对定量推理问题的预测最终答案是否与真实答案匹配来给予奖励 (Guo et al., 2025)。该方法足以使前沿大语言模型在主要评估最终答案的数学竞赛 (如 AIME 和 HMMT) 中达到性能上限。然而,这种奖励机制存在两个根本性局限。首先,它作为推理正确性的代理指标并不可靠——模型可能通过有缺陷的逻辑或侥幸的错误得出正确答案。其次,它不适用于定理证明任务,因为此类问题可能不需要输出数值型最终答案,且严谨的推导过程才是主要目标。

\*Para 2:\* 因此,仅基于此类最终答案奖励在定量推理问题上训练的大语言模型,仍会频繁生成数学上无效或逻辑不一致的自然语言证明。此外,这种训练方法并未自然培养模型验证证明有效性的能力——它们表现出较高的假阳性率,即使证明中存在明显的逻辑缺陷,也常常错误地判定其为有效。

\*Para 3:\* 自然语言定理证明中生成与验证之间的鸿沟 (generation-verification gap) 阻碍了进一步的提升。为解决这一问题,我们提出在大语言模型中开发证明验证能力。我们的方法基于以下几项关键观察: 人类即使在没有参考解答的情况下也能识别证明中的问题——这是解决开放性问题的关键能力。当经过大规模验证努力仍无法识别出问题时,该证明有效的可能性更高。识别有效问题所需的努力可作为证明质量的代理指标,进而可用于优化证明生成。

\*Para 4:\* 我们相信,大语言模型可以被训练为在无参考解答的情况下识别证明问题。此类验证器将实现一个迭代优化循环: (1) 利用验证反馈优化证明生成; (2) 扩展验证计算资源以自动标注难以验证的新证明,从而生成用于改进验证器本身的训练数据; (3) 利用增强后的验证器进一步优化证明生成。此外,可靠的证明验证器使我们能够教导证明生成器像验证器一样评估证明。这使得证明生成器能够迭代地完善其证明,直到无法再识别或解决任何问题为止。本质上,我们

传统的用于数学推理的强化学习 (RL) 方法,主要根据大语言模型 (LLMs) 对定量推理问题的预测最终答案是否与真实答案匹配来给予奖励 (Guo et al., 2025)。该方法足以使前沿大语言模型在主要评估最终答

案的数学竞赛（如 AIME 和 HMMT）中达到性能上限。然而，这种奖励机制存在两个根本性局限。首先，它作为推理正确性的代理指标并不可靠——模型可能通过有缺陷的逻辑或侥幸的错误得出正确答案。其次，它不适用于定理证明任务，因为此类问题可能不需要输出数值型最终答案，且严谨的推导过程才是主要目标。因此，仅基于此类最终答案奖励在定量推理问题上训练的大语言模型，仍会频繁生成数学上无效或逻辑不一致的自然语言证明。此外，这种训练方法并未自然培养模型验证证明有效性的能力——它们表现出较高的假阳性率，即使证明中存在明显的逻辑缺陷，也常常错误地判定其为有效。自然语言定理证明中生成与验证之间的差距（generation-verification gap）阻碍了进一步的提升。为解决这一问题，我们提出在大语言模型中开发证明验证能力。我们的方法基于以下几项关键观察：人类即使在无参考解答的情况下也能识别证明中的问题——这是解决开放性问题的关键能力。当经过大规模验证努力仍无法识别出问题时，该证明有效的可能性更高。识别有效问题所需的努力可作为证明质量的代理指标，进而可用于优化证明生成。我们相信，大语言模型可以被训练为在无参考解答的情况下识别证明问题。此类验证器将实现一个迭代优化循环：（1）利用验证反馈优化证明生成；（2）扩展验证计算规模以自动标注难以验证的新证明，从而生成用于改进验证器本身的训练数据；（3）利用增强后的验证器进一步优化证明生成。此外，可靠的证明验证器使我们能够教导证明生成器像验证器一样评估证明。这使得证明生成器能够迭代地完善其证明，直到无法再识别或解决任何问题为止。本质上，我们

\*Output Generation\* (matches the final refined version) “1. 引言 传统的用于数学推理的强化学习 (RL) 方法，主要根据大语言模型 (LLMs) 对定量推理问题的预测最终答案是否与真实答案匹配来给予奖励 (Guo et al., 2025)。该方法足以使前沿大语言模型在主要评估最终答案的数学竞赛（如 AIME 和 HMMT）中达到性能上限。然而，这种奖励机制存在两个根本性局限。首先，它作为推理正确性的代理指标并不可靠——模型可能通过有缺陷的逻辑或侥幸的错误得出正确答案。其次，它不适用于定理证明任务，因为此类问题可能不需要输出数值型最终答案，且严谨的推导过程才是主要目标。因此，仅基于此类最终答案奖励在定量推理问题上训练的大语言模型，仍会频繁生成数学上无效或逻辑不一致的自然语言证明。此外，这种训练方法并未自然培养模型验证证明有效性的能力——它们表现出较高的假阳性率，即使证明中存在明显的逻辑缺陷，也常常错误地判定其为有效。自然语言定理证明中生成与验证之间的差距 (generation-verification gap) 阻碍了进一步的提升。为解决这一问题，我们提出在大语言模型中开发证明验证能力。我们的方法基于以下几项关键观察：人类即使在无参考解答的情况下也能识别证明

中的问题——这是解决开放性问题的关键能力。当经过扩展的验证努力仍无法识别出问题时，该证明有效的可能性更高。识别真实存在的问题所需的努力可作为证明质量的代理指标，进而可用于优化证明生成。我们相信，大语言模型可以被训练为在无参考解答的情况下识别证明问题。此类验证器将实现一个迭代优化循环：（1）利用验证反馈优化证明生成；（2）扩展验证计算规模以自动标注难以验证的新证明，从而生成用于改进验证器本身的训练数据；（3）利用增强后的验证器进一步优化证明生成。此外，可靠的证明验证器使我们能够教导证明生成器像验证器一样评估证明。这使得证明生成器能够迭代地完善其证明，直到无法再识别或解决任何问题为止。本质上，我们” Matches perfectly. Output matches exactly. Proceeds.

使模型显式感知其奖励函数，并使其能够通过深思熟虑的推理而非盲目试错来最大化该奖励。基于 DeepSeek-V3.2-Exp-Base (DeepSeek-AI, 2025)，我们开发了 DeepSeekMath-V2，这是一款针对自然语言定理证明进行优化的大语言模型，具备可自我验证的数学推理能力。该模型能够评估并迭代优化其自身的证明，在包括2025年IMO和2024年CMO在内的顶尖高中数学竞赛中取得了金牌级表现。在2024年普特南 (Putnam) 本科生数学竞赛中，该模型取得118/120分，超越了人类参赛者的最高得分90分。Here's a thinking process:

118/120, exceeding the highest score of 90 obtained by human participants." I will match the line breaks exactly as requested, or just output as a coherent paragraph. The prompt says "保持原文结构和格式", so I'll keep the line breaks. Let's align line by line: 使模型显式感知其奖励函数，并使其能够通过深思熟虑的推理而非盲目试错来最大化该奖励。基于 DeepSeek-V3.2-Exp-Base (DeepSeek-AI, 2025)，我们开发了 DeepSeekMath-V2，这是一款针对自然语言定理证明进行优化的大语言模型，具备可自我验证的数学推理能力。该模型能够评估并迭代优化其自身的证明，在包括2025年IMO和2024年CMO在内的顶尖高中数学竞赛中取得了金牌级表现。在2024年普特南 (Putnam) 本科生数学竞赛中，该模型取得118/120分，超越了人类参赛者的最高得分90分。Actually, the original line breaks are just due to PDF/column width. I'll output it as a single paragraph or keep the breaks. I'll keep the breaks to be safe. Let's refine the translation to be strictly academic: 使模型显式感知其奖励函数，并使其能够通过深思熟虑的推理而非盲目试错来最大化该奖励。基于 DeepSeek-V3.2-Exp-Base (DeepSeek-AI, 2025)，我们开发了 DeepSeekMath-V2，这是一款针对自然语言定理证明进行优化的大语言模型，展现出可自我验证的数学推理能力。该模型能够评估并迭代改进

其自身的证明，在包括2025年IMO和2024年CMO在内的顶尖高中数学竞赛中取得了金牌级表现。在2024年普特南 (Putnam) 本科生数学竞赛中，该模型得分118/120，超越了人类参赛者的最高分90分。

## 2. Method

我们制定了用于证明评估的高层评分标准  $I_v$  (见附录 A.2)，旨在训练验证器依据这些标准对证明进行评估，从而模拟数学专家的评估流程。具体而言，给定问题  $X$  和证明  $Y$ ，验证器  $\pi_\phi(\cdot|X, Y, I_v)$  被设计为生成一份证明分析，首先总结所发现的问题 (如有)，随后根据三个等级进行评分：1 分对应完整且严谨的证明，所有逻辑步骤均论证充分；0.5 分对应整体逻辑正确但存在细微错误或遗漏细节的证明；0 分对应存在致命逻辑错误或关键漏洞的根本性缺陷证明。

构建冷启动强化学习数据

我们通过以下流程构建初始训练数据：Here's a thinking process:

我们制定了用于证明评估的高层评估准则  $I_v$  (见附录 A.2)，旨在训练验证器依据这些准则对证明进行评估，从而模拟数学专家的评估流程。具体而言，给定问题  $X$  和证明  $Y$ ，验证器  $\pi_\phi(\cdot|X, Y, I_v)$  被设计为生成一份证明分析，首先总结所发现的问题 (如有)，随后根据三个等级进行评分：1 分对应完整且严谨的证明，所有逻辑步骤均论证充分；0.5 分对应整体逻辑正确但存在细微错误或遗漏细节的证明；0 分对应存在致命逻辑错误或关键漏洞的根本性缺陷证明。

构建冷启动强化学习数据  
我们通过以下流程构建初始训练数据：

[Paragraph] 构建冷启动强化学习数据

我们制定了用于证明评估的高层评估准则  $I_v$  (见附录 A.2)，旨在训练验证器依据这些准则对证明进行评估，从而模拟数学专家的评估流程。具体而言，给定问题  $X$  和证明  $Y$ ，验证器  $\pi_\phi(\cdot|X, Y, I_v)$  被设计为生成一份证明分析，首先总结所发现的问题 (如有)，随后根据三个等级进行评分：1 分对应完整且严谨的证明，所有逻辑步骤均论证充分；0.5 分对应整体逻辑正确但存在细微错误或遗漏细节的证明；0 分对应存在致命逻辑错误或关键漏洞的根本性缺陷证明。

构建冷启动强化学习数据

我们通过以下流程构建初始训练数据：

我们制定了用于证明评估的高层评分标准  $I_v$  (见附录 A.2)，旨在训练验证器依据这些标准对证明进行评估，从而模拟数学专家的评估流程。具体而言，给定问题  $X$  和证明  $Y$ ，验证器  $\pi_\phi(\cdot|X, Y, I_v)$  被设计为生成一份证明分析，首先总结所发现的问题 (如有)，随后根据三个等级进行评分：1 分对应完整且严谨的证明，所有逻辑步骤均论证充分；0.5 分对应整体逻辑正确但存在细微错误或遗漏细节的证明；0 分对应存在致命逻辑错误或关键漏洞的根本性缺陷证明。

构建冷启动强化学习数据

我们通过以下流程构建初始训练数据：” Matches perfectly. Output matches requirement. Proceeds.

1. We crawled problems from Art of Problem Solving (AoPS) contests, prioritizing math

Combine: 1. 我们从 Art of Problem Solving (AoPS) 竞赛中爬取了题目，优先选取数学奥林匹克竞赛、国家队选拔测试以及2010年之后明确要求证明的题目，共计17,503道。该题集记为  $D_p$ 。

Refined: 1. 我们从 Art of Problem Solving (AoPS) 竞赛中爬取了题目，优先选取数学奥林匹克竞赛、国家队选拔测试以及2010年之后明确要求证明的题目，共计17,503道。该题集记为  $D_p$ 。 One minor adjustment for flow and academic tone: "1. 我们从 Art of Problem Solving (AoPS) 竞赛平台爬取了题目，优先选取数学奥林匹克竞赛、国家队选拔测试以及2010年之后明确要求证明的题目，共计17,503道。该题集记为  $D_p$ 。" (Added "平台" for clarity, but maybe not necessary. I'll stick closer to original: "竞赛中")

2. We generated candidate proofs using a variant of DeepSeek-V3.2-Exp-Thinking. As this

Refined: 2. 我们使用 DeepSeek-V3.2-Exp-Thinking 的一个变体生成了候选证明。由于该模型并未针对定理证明进行优化，且倾向于生成简洁但易出错的输出，我们提示其在多轮迭代中逐步完善证明，以提升其全面性与严谨性。

Draft: 2. 我们使用 DeepSeek-V3.2-Exp-Thinking 的一个变体生成了候选证明。由于该模型并未针对定理证明进行优化，且倾向于生成简洁但易出错的输出，我们提示其在多轮迭代中逐步完善证明，以提升其全面性与严谨性。 One minor adjustment: "prompted it" in LLM context is usually translated as "通过提示词引导" or simply "提示". "提示其" is fine. "iteratively refine its proofs over multiple rounds" -> "在多轮交互中迭代优化其证明". Let's stick to: "我们提示其在多轮迭代中逐步完善证明". Actually, "多轮迭代" covers "over multiple rounds" and "iteratively". To be precise: "我们提示其在多轮过程中迭代优化证明". Or "我们引导其在多轮迭代中逐步完善证明". I'll go with: "我们提示其在多轮迭代中逐步完善证明".

Let's verify exact wording: "2. 我们使用 DeepSeek-V3.2-Exp-Thinking 的一个变体生成了候选证明。由于该模型并未针对定理证明进行优化，且倾向于生成简洁但易出错的输出，我们提示其在多轮迭代中逐步完善证明，以提升其全面性与严谨性。"

[Output Generation] (matches the final refined version) -> `2. 我们采用 DeepSeek-V3.2-Exp-Thinking

的一个变体生成了候选证明。鉴于该模型未针对定理证明进行优化，且其输出往往简洁但易出错，我们通过提示引导其在多轮迭代中逐步完善证明，以提升其全面性与严谨性。`All good.

\*(Note: I will output exactly this string without any extra text as instructed.)\* Final Output Generation. `2. 我们采用 DeepSeek-V3.2-Exp-Thinking 的一个变体生成了候选证明。鉴于该模型未针对定理证明进行优化，且其输出往往简洁但易出错，我们通过提示引导其在多轮迭代中逐步完善证明，以提升其全面性与严谨性。` Done.

3. We randomly sampled proofs across diverse problem types (e.g., algebra and number

该过程生成了一个初始强化学习数据集  $D_v = \{(X_i, Y_i, s_i)\}$ ，其中每个样本包含一个问题  $X_i$ 、一个证明  $Y_i$  以及一个总体证明分数  $s_i \in \{0, 0.5, 1\}$ 。强化学习目标。基于在数学和代码相关推理数据上进行监督微调的 DeepSeek-V3.2-Exp-SFT 版本，我们使用强化学习训练模型以生成证明分析，该训练采用两个奖励组件：格式奖励  $R_{\text{format}}$ ：一个指示函数，用于强制模型同时生成已识别问题的摘要和证明分数。该函数通过检查最终回复是否包含关键短语“Here is my evaluation of the solution:”以及在“Based on my evaluation, the final overall score should be:”之后是否包含位于  $\boxed{\}$  中的分数来实现。分数奖励  $R_{\text{score}}$ ：基于预测分数  $s'_i$  与标注分数  $s_i$  之间接近程度的奖励： $R_{\text{score}}(s'_i, s_i) = 1 - |s'_i - s_i|$  (1) 训练验证器的强化学习目标为： $\max_{\pi} E_{(X_i, Y_i, s_i) \sim D_v} [R_{\text{format}}(V'_i) \cdot R_{\text{score}}(s'_i, s_i)]$  (2) 其中  $V'_i$  表示验证器的最终回复， $s'_i$  为从中提取的证明分数。

第 2.1.1 节所述的方法通过强化学习训练证明验证，以使预测的证明分数与专家标注对齐，但并未对识别出的问题本身提供直接监督。这导致了一个关键缺陷：在训练期间评估有缺陷的证明（即  $s_i < 1$ ）时，验证器可以通过预测正确的分数同时虚构不存在的问题来获得全额奖励，从而削弱其可信度。为解决该问题，我们引入了元验证（meta-verification）：一种二次评估过程，用于评估验证器识别出的问题是否确实存在，以及这些问题是否根据评估标准  $I_v$  在逻辑上合理支撑了预测的证明分数。完整的元验证标准  $I_{mv}$  详见附录 A.3。我们使用强化学习训练了一个专用的元验证器来执行此评估。通过将元验证器的反馈纳入验证器的训练中，我们可以提高验证器问题识别的忠实度。元验证器训练过程 Here's a thinking process:

该过程生成了一个初始强化学习数据集  $D_v = \{(X_i, Y_i, s_i)\}$ ，其中每个样本包含一个问题  $X_i$ 、一个证明  $Y_i$  以及一个总体证明分数  $s_i \in \{0, 0.5, 1\}$ 。

**\*\*强化学习目标。\*\***

基于在数学和代码相关推理数据上进行监督微调的 DeepSeek-V3.2-Exp-SFT 版本，我们使用强化学习训练模型以生成证明分析，该训练采用两个奖励组件：格式奖励  $R_{\text{format}}$ ：一个指示函数，用于强制模型同时生成已识别问题的摘要和证明分数。其通过检查最终回复是否包含关键短语“Here is my evaluation of the solution:”（此处为我对解答的评估：）以及在“Based on my evaluation, the final overall score should be:”（根据我的评估，最终总体分数应为：）之后是否包含位于  $\boxed{\}$  中的分数来实现。分数奖励  $R_{\text{score}}$ ：基于预测分数  $s'_i$  与标注分数  $s_i$  之间接近程度的奖励： $R_{\text{score}}(s'_i, s_i) = 1 - |s'_i - s_i|$  (1) 训练验证器的强化学习目标为： $\max_{\pi} E_{(X_i, Y_i, s_i) \sim D_v} [R_{\text{format}}(V'_i) \cdot R_{\text{score}}(s'_i, s_i)]$  (2) 其中  $V'_i$  表示验证器的最终回复， $s'_i$  为从中提取的证明分数。

**\*\*2.1.2. 引入元验证以审查证明分析\*\*** 第 2.1.1 节所述的方法通过强化学习训练证明验证，以使预测的证明分数与专家标注对齐，但并未对识别出的问题本身提供直接监督。这导致了一个关键缺陷：在训练期间评估有缺陷的证明（即  $s_i < 1$ ）时，验证器可以通过预测正确的分数同时捏造（幻觉）出不存在的问题来获得全额奖励，从而削弱其可信度。为解决该问题，我们引入了元验证（meta-verification）：一种二次评估过程，用于评估验证器识别出的问题是否确实存在，以及这些问题是否根据评估标准  $I_v$  在逻辑上合理支撑了预测的证明分数。完整的元验证标准  $I_{mv}$  详见附录 A.3。我们使用强化学习训练了一个专用的元验证器来执行此评估。通过将元验证器的反馈纳入验证器的训练中，我们可以提高验证器问题识别的忠实度。

**\*\*元验证器训练过程\*\***

该过程生成了一个初始强化学习数据集  $D_v = \{(X_i, Y_i, s_i)\}$ ，其中每个样本包含一个问题  $X_i$ 、一个证明  $Y_i$  以及一个总体证明分数  $s_i \in \{0, 0.5, 1\}$ 。强化学习目标。基于在数学和代码相关推理数据上进行监督微调的 DeepSeek-V3.2-Exp-SFT 版本，我们使用强化学习训练模型以生成证明分析，该训练采用两个奖励组件：格式奖励  $R_{\text{format}}$ ：一个指示函数，用于强制模型同时生成已识别问题的摘要和证明分数。该函数通过检查最终回复是否包含关键短语“Here is my evaluation of the solution:”以及在“Based on my evaluation, the final overall score should be:”之后是否包含位于  $\boxed{\}$  中的分数来实现。分数奖励  $R_{\text{score}}$ ：基于预测分数  $s'_i$  与标注分数  $s_i$  之间接近程度的奖励： $R_{\text{score}}(s'_i, s_i) = 1 - |s'_i - s_i|$  (1)

训练验证器的强化学习目标为： $\max_{\pi} \mathbb{E}_{(X_i, Y_i, s_i) \sim D_v} (V'_i(s'_i) \cdot R_{\text{format}}(V'_i) \cdot R_{\text{score}}(s'_i, s_i))$  (2) 其中  $V'_i$  表示验证器的最终回复， $s'_i$  为从中提取的证明分数。

第 2.1.1 节所述的方法通过强化学习训练证明验证，以使预测的证明分数与专家标注对齐，但并未对识别出的问题本身提供直接监督。这导致了一个关键缺陷：在训练期间评估有缺陷的证明（即  $s_i < 1$ ）时，验证器可以通过预测正确的分数同时虚构不存在的问题来获得全额奖励，从而削弱其可信度。为解决该问题，我们引入了元验证（meta-verification）：一种二次评估过程，用于评估验证器识别出的问题是否确实存在，以及这些问题是否根据评估标准  $I_v$  在逻辑上合理支撑了预测的证明分数。完整的元验证标准  $I_{mv}$  详见附录 A.3。我们使用强化学习训练了一个专用的元验证器来执行此评估。通过将元验证器的反馈纳入验证器的训练中，我们可以提高验证器问题识别的忠实度。

元验证器训练过程

第 2.1.1 节所述的方法通过强化学习训练证明验证，以使预测的证明分数与专家标注对齐，但并未对识别出的问题本身提供直接监督。这导致了一个关键缺陷：在训练期间评估有缺陷的证明（即  $s_i < 1$ ）时，验证器可以通过预测正确的分数同时虚构不存在的问题来获得全额奖励，从而削弱其可信度。为解决该问题，我们引入了元验证（meta-verification）：一种二次评估过程，用于评估验证器识别出的问题是否确实存在，以及这些问题是否根据评估标准  $I_v$  在逻辑上合理支撑了预测的证明分数。完整的元验证标准  $I_{mv}$  详见附录 A.3。我们使用强化学习训练了一个专用的元验证器来执行此评估。通过将元验证器的反馈纳入验证器的训练中，我们可以提高验证器问题识别的忠实度。

元验证器训练过程"

[Output] -> (Proceeds) "3. 我们跨多种问题类型（例如代数与数论）随机采样了证明，并让数学专家根据上述评估标准对每个证明进行评分。该过程生成了一个初始强化学习数据集  $D_v = \{(X_i, Y_i, s_i)\}$ ，其中每个样本包含一个问题  $X_i$ 、一个证明  $Y_i$  以及一个总体证明分数  $s_i \in \{0, 0.5, 1\}$ 。强化学习目标。基于在数学和代码相关推理数据上进行监督微调的 DeepSeek-V3.2-Exp-SFT 版本，我们使用强化学习训练模型以生成证明分析，该训练采用两个奖励组件：  
格式奖励  $R_{\text{format}}$ ：一个指示函数，用于强制模型同时生成已识别问题的摘要和证明分数。该函数通过检查最终回复是否包含关键词语“Here is my evaluation of the solution:”以及在“Based on my evaluation, the final overall score should be:”之后是否包含位于  $\boxed{\quad}$  中的分数来实现。  
分数奖励  $R_{\text{score}}$ ：基于预测分数  $s'_i$  与标注分数  $s_i$

之间接近程度的奖励： $R_{\text{score}}(s'_i, s_i) = 1 - |s'_i - s_i|$  (1) 训练验证器的强化学习目标为： $\max_{\pi} \mathbb{E}_{(X_i, Y_i, s_i) \sim D_v} (V'_i(s'_i) \cdot R_{\text{format}}(V'_i) \cdot R_{\text{score}}(s'_i, s_i))$  (2) 其中  $V'_i$  表示验证器的最终回复， $s'_i$  为从中提取的证明分数。

第 2.1.1 节所述的方法通过强化学习训练证明验证，以使预测的证明分数与专家标注对齐，但并未对识别出的问题本身提供直接监督。这导致了一个关键缺陷：在训练期间评估有缺陷的证明（即  $s_i < 1$ ）时，验证器可以通过预测正确的分数同时虚构不存在的问题来获得全额奖励，从而削弱其可信度。为解决该问题，我们引入了元验证（meta-verification）：一种二次评估过程，用于评估验证器识别出的问题是否确实存在，以及这些问题是否根据评估标准  $I_v$  在逻辑上合理支撑了预测的证明分数。完整的元验证标准  $I_{mv}$  详见附录 A.3。我们使用强化学习训练了一个专用的元验证器来执行此评估。通过将元验证器的反馈纳入验证器的训练中，我们可以提高验证器问题识别的忠实度。

元验证器训练过程” Matches perfectly. Output matches this. Proceeds.

2. Mathematical experts scored the quality of verifier responses according to  $I_{mv}$ , creating

2. 数学专家根据  $I_{mv}$  对验证器回复的质量进行评分，构建数据集  $D_{mv} = \{(X_i, Y_i, V_i, ms_i)\}$ ，其中  $V_i$  是对证明  $Y_i$  的分析， $ms_i \in \{0, 0.5, 1\}$  为专家标注的质量分数。

3. We trained a meta-verifier  $\pi_{\eta}(\cdot | X, Y, V, I_{mv})$  to analyze the verifier's proof analysis  $V$ . The

$R_V = R_{\text{format}} \cdot R_{\text{score}} \cdot R_{\text{meta}}$  (3) 其中  $R_{\text{meta}}$  为元验证器给出的质量分数。我们在验证数据集  $D_v$  与元验证数据集  $D_{mv}$  上对增强后的验证器进行了训练，并在  $D_{mv}$  上采用了与训练元验证器相同的奖励机制。所得模型能够同时执行证明验证与元验证任务。在  $D_v$  的验证集上，验证器证明分析的平均质量分数（经元验证器评估）从 0.85 提升至 0.96，同时在证明分数预测上保持了相同的准确率。

以验证器  $\pi_{\phi}$  作为生成式奖励模型，我们采用如下强化学习目标训练证明生成器  $\pi_{\theta}(\cdot | X)$ ： $\max_{\pi} \mathbb{E}_{X_i \sim D_p, Y_i \sim \pi_{\theta}(\cdot | X_i)} [R_Y]$  (4) 其中  $R_Y$  为由  $\pi_{\phi}(\cdot | X_i, Y_i, I_v)$  生成的证明分数。

当证明生成器无法一次性生成完全正确的证明时——这在 IMO 和 CMO 等竞赛的挑战性题目中十分常见——迭代验证与修正可有效提升结果。该过程涉及使用外部验证器对证明进行分析，并提示生成器针对已识别的问题进行修正。然而，我们观察到一个关键局限性：当要求生成器一次性完成证明生成与自我分析时，即

便外部验证器能轻易识别出缺陷，生成器仍倾向于断言其正确性。换言之，尽管生成器能够依据外部反馈对证明进行修正，却无法像专用验证器那样以同等严谨的标准评估自身输出。这一观察促使我们为证明生成器赋予真正的验证能力。在训练过程中，我们提示生成器  $\pi_\theta$  先生成证明  $Y$ ，随后生成一份自我分析  $Z$ ，该分析遵循与验证器相同的格式与评分标准  $l_v$ （详见附录

A.1）。我们将自我分析中预测的证明分数记为  $s'$ 。为确保自我评估的准确性，我们使用验证器  $\pi_\phi$  对两个组成部分进行评估：证明 Here's a thinking process:

\*Draft:\* 3. 我们训练了一个元验证器  $\pi_\eta(\cdot|X, Y, V, l_{mv})$  来分析验证器的证明分析  $V$ 。该

\*Refinement:\* 3. 我们训练了一个元验证器  $\pi_\eta(\cdot|X, Y, V, l_{mv})$  以分析验证器的证明分析结果  $V$ 。该

\*Draft:\* 元验证器生成分析中发现的问题摘要，随后给出一个质量分数，衡量验证器分析的准确性和合理性。

\*Refinement:\* 元验证器会生成一份针对分析本身所发现问题的摘要，随后给出一个质量分数，用于衡量验证器分析的准确性与合理性。

\*Draft:\* RL目标遵循与验证器训练相同的结构，包含格式和分数奖励。

\*Refinement:\* 强化学习 (RL) 目标遵循与验证器训练相同的结构，包含格式奖励与分数奖励。

\*Draft:\* 使用训练好的元验证器  $\pi_\eta$ ，我们通过将元验证反馈整合到奖励函数中来增强验证器训练：

\*Refinement:\* 利用训练好的元验证器  $\pi_\eta$ ，我们通过将元验证反馈整合至奖励函数中，从而增强验证器的训练：

\*Draft:\* 其中  $R_{meta}$  是元验证器的质量分数。

\*Refinement:\* 其中  $R_{meta}$  为元验证器给出的质量分数。

\*Draft:\* 我们在验证数据集  $D_v$  和元验证数据集  $D_{mv}$  上训练了增强后的验证器，在  $D_{mv}$  上使用与训练元验证器相同的奖励机制。

\*Refinement:\* 我们在验证数据集  $D_v$  与元验证数据集  $D_{mv}$  上对增强后的验证器进行了训练，并在  $D_{mv}$  上采用了与训练元验证器相同的奖励机制。

\*Draft:\* 得到的模型可以执行证明验证和元验证任务。

\*Refinement:\* 所得模型能够同时执行证明验证与元验证任务。

\*Draft:\* 在  $D_v$  的验证集上，验证器证明分析的平均质量分数（由元验证器评估）从 0.85 提高到 0.96，同时保持了相同的证明分数预测准确率。

\*Refinement:\* 在  $D_v$  的验证划分上，验证器证明分析的平均质量分数（经元验证器评估）从 0.85 提升至 0.96，同时在证明分数预测上保持了相同的准确率。

\*Draft:\* 2.2.1. 训练定理证明生成器

\*Draft:\* 以验证器  $\pi_\phi$  作为生成奖励模型，我们使用以下RL目标训练证明生成器  $\pi_\theta(\cdot|X)$ ：

\*Refinement:\* 以验证器  $\pi_\phi$  作为生成式奖励模型，我们采用如下强化学习目标训练证明生成器  $\pi_\theta(\cdot|X)$ ：

\*Draft:\* 2.2.2. 通过自我验证增强推理

\*Draft:\* 当证明生成器无法一次性生成完全正确的证明时——这在IMO和CMO等竞赛的挑战性问题中很常见——迭代验证和细化可以改善结果。

\*Refinement:\* 当证明生成器无法一次性生成完全正确的证明时——这在IMO（国际数学奥林匹克）和CMO（中国数学奥林匹克）等竞赛的挑战性题目中十分常见——迭代验证与修正可有效提升结果。

\*Draft:\* 这涉及使用外部验证器分析证明，并提示生成器解决发现的问题。

\*Refinement:\* 该过程涉及使用外部验证器对证明进行分析，并提示生成器针对已识别的问题进行修正。

\*Draft:\* 然而，我们观察到一个关键限制：当提示生成器一次性生成并分析自己的证明时，即使外部验证器很容易发现缺陷，生成器也倾向于声称正确。

\*Refinement:\* 然而，我们观察到一个关键局限性：当要求生成器一次性完成证明生成与自我分析时，即便外部验证器能轻易识别出缺陷，生成器仍倾向于断言其正确性。

\*Draft:\* 换句话说，虽然生成器可以根据外部反馈细化证明，但它无法像专用验证器那样严格评估自己的工作。

\*Refinement:\* 换言之，尽管生成器能够依据外部反馈对证明进行修正，却无法像专用验证器那样以同等严谨的标准评估自身输出。

\*Draft:\* 这一观察促使我们赋予证明生成器真正的验证能力。

\*Refinement:\* 这一观察促使我们为证明生成器赋予真正的验证能力。

\*Draft:\* 在训练期间，我们提示生成器  $\pi_\theta$  生成证明  $Y$ ，随后生成自我分析  $Z$ ，遵循与验证器相同的格式和标准  $l_v$ （见附录 A.1）。

\*Refinement:\* 在训练过程中，我们提示生成器  $\pi_\theta$  先生成证明  $Y$ ，随后生成一份自我分析  $Z$ ，该分析遵循与验证器相同的格式与评分标准  $l_v$  (详见附录 A.1)。

\*Draft:\* 我们将自我分析中预测的证明分数记为  $s'$ 。

\*Refinement:\* 我们将自我分析中预测的证明分数记为  $s'$ 。

\*Draft:\* 为了确保忠实的自我评估，我们使用验证器  $\pi_\phi$  来评估两个部分：证明

\*Refinement:\* 为了确保自我评估的忠实性，我们使用验证器  $\pi_\phi$  对两个组成部分进行评估：证明

Let's assemble and verify carefully. "3. 我们训练了一个元验证器  $\pi_\eta(\cdot|X, Y, V, l_{mv})$  以分析验证器的证明分析结果  $V$ 。该元验证器会生成一份针对分析本身所发现问题的摘要，随后给出一个质量分数，用于衡量验证器分析的准确性与合理性。强化学习 (RL) 目标遵循与验证器训练相同的结构，包含格式奖励与分数奖励。利用训练好的元验证器  $\pi_\eta$ ，我们通过将元验证反馈整合至奖励函数中，从而增强验证器的训练： $R_V = R_{format} \cdot R_{score} \cdot R_{meta}$  (3) 其中  $R_{meta}$  为元验证器给出的质量分数。我们在验证数据集  $D_v$  与元验证数据集  $D_{mv}$  上对增强后的验证器进行了训练，并在  $D_{mv}$  上采用了与训练元验证器相同的奖励机制。所得模型能够同时执行证明验证与元验证任务。在  $D_v$  的验证划分上，验证器证明分析的平均质量分数 (经元验证器评估) 从 0.85 提升至 0.96，同时在证明分数预测上保持了相同的准确率。

以验证器  $\pi_\phi$  作为生成式奖励模型，我们采用如下强化学习目标训练证明生成器  $\pi_\theta(\cdot|X) : \max_{\pi_\theta} E[X_i \sim D_p, Y_i \sim \pi_\theta(\cdot|X_i)] [R_Y]$  (4) 其中  $R_Y$  为由  $\pi_\phi(\cdot|X_i, Y_i, l_v)$  生成的证明分数。

当证明生成器无法一次性生成完全正确的证明时——这在 IMO 和 CMO 等竞赛的挑战性题目中十分常见——迭代验证与修正可有效提升结果。该过程涉及使用外部验证器对证明进行分析，并提示生成器针对已识别的问题进行修正。然而，我们观察到一个关键局限性：当要求生成器一次性完成证明生成与自我分析时，即便外部验证器能轻易识别出缺陷，生成器仍倾向于断言其正确性。换言之，尽管生成器能够依据外部反馈对证明进行修正，却无法像专用验证器那样以同等严谨的标准评估自身输出。这一观察促使我们为证明生成器赋予真正的验证能力。在训练过程中，我们提示生成器  $\pi_\theta$  先生成证明  $Y$ ，随后生成一份自我分析  $Z$ ，该分析遵循与验证器相同的格式与评分标准  $l_v$  (详见附录

A.1)。我们将自我分析中预测的证明分数记为  $s'$ 。为确保自我评估的忠实性，我们使用验证器  $\pi_\phi$  对两个组成部分进行评估：证明"

$R_V = R_{format} \cdot R_{score} \cdot R_{meta}$  (3) 其中  $R_{meta}$  为元验证器给出的质量分数。我们在验证数据集  $D_v$  与元验证数据集  $D_{mv}$  上对增强后的验证器进行了训练，并在  $D_{mv}$  上采用了与训练元验证器相同的奖励机制。所得模型能够同时执行证明验证与元验证任务。在  $D_v$  的验证集上，验证器证明分析的平均质量分数 (经元验证器评估) 从 0.85 提升至 0.96，同时在证明分数预测上保持了相同的准确率。

以验证器  $\pi_\phi$  作为生成式奖励模型，我们采用如下强化学习目标训练证明生成器  $\pi_\theta(\cdot|X) : \max_{\pi_\theta} E[X_i \sim D_p, Y_i \sim \pi_\theta(\cdot|X_i)] [R_Y]$  (4) 其中  $R_Y$  为由  $\pi_\phi(\cdot|X_i, Y_i, l_v)$  生成的证明分数。

当证明生成器无法一次性生成完全正确的证明时——这在 IMO 和 CMO 等竞赛的挑战性题目中十分常见——迭代验证与修正可有效提升结果。该过程涉及使用外部验证器对证明进行分析，并提示生成器针对已识别的问题进行修正。然而，我们观察到一个关键局限性：当要求生成器一次性完成证明生成与自我分析时，即便外部验证器能轻易识别出缺陷，生成器仍倾向于断言其正确性。换言之，尽管生成器能够依据外部反馈对证明进行修正，却无法像专用验证器那样以同等严谨的标准评估自身输出。这一观察促使我们为证明生成器赋予真正的验证能力。在训练过程中，我们提示生成器  $\pi_\theta$  先生成证明  $Y$ ，随后生成一份自我分析  $Z$ ，该分析遵循与验证器相同的格式与评分标准  $l_v$  (详见附录 A.1)。我们将自我分析中预测的证明分数记为  $s'$ 。为确保自我评估的准确性，我们使用验证器  $\pi_\phi$  对两个组成部分进行评估：证明

$Y$  获得分数  $R_Y = s$ ，自我分析  $Z$  获得元验证分数  $R_{meta}(Z) = ms$ 。  $R_{score}(s', s)$  用于奖励准确的自我评估。我们设定  $\alpha = 0.76$ ，  $\beta = 0.24$ 。该奖励结构产生了以下激励效应：

如实承认错误比虚假声称正确能获得更高的奖励。最高奖励来自于生成正确的证明并准确识别其严谨性。对于证明生成器而言，获取高奖励的有效策略是在最终输出前尽可能多地发现并解决问题。证明验证器与生成器之间形成了一个协同循环：验证器能够提升生成器的性能，而随着生成器的改进，它又会生成新的证明，从而对验证器当前的能力提出挑战。这些具有挑战性的案例——即验证器可能无法在单次尝试中识别出问题的案例——将成为提升验证器自身能力的宝贵训练数据。为了重新训练并改进验证器，我们需要为新生成的证明提供带有正确性标签的数据。尽管人工标注方法直接明了，但随着问题难度增加和错误变得更为隐蔽，其

耗时也会显著上升。为提高标注效率，我们为每个证明生成了多个验证器分析结果，以凸显潜在问题供人工审查。

2. Reviewing the verifier's identified issues is exactly meta-verification, which is easier than

从头识别问题。元验证在供大语言模型掌握方面也具有更高的样本效率。基于上述观察，我们开发了以下自动化标注流程：Combine: 从头识别问题。元验证在供大语言模型掌握方面也具有更高的样本效率。基于上述观察，我们开发了以下自动化标注流程：Refined: 从头识别问题。元验证在供大语言模型掌握方面也具有更高的样本效率。基于上述观察，我们设计了以下自动化标注流程：“……从头识别问题。此外，大语言模型掌握元验证的样本效率也更高。基于上述观察，我们开发了如下自动化标注流程：”“从头识别问题。元验证在供大语言模型掌握方面也更具样本效率。基于上述观察，我们设计了以下自动化标注流程：” Check "sample-efficient for LLMs to master": This means LLMs need fewer samples to learn/master meta-verification. So: “大语言模型学习掌握元验证的样本效率也更高。” or “元验证也更具样本效率，更易于大语言模型掌握。” I'll go with: “元验证在供大语言模型掌握方面也具有更高的样本效率。” 从头识别问题。元验证在供大语言模型掌握方面也具有更高的样本效率。

基于上述观察，我们设计了以下自动化标注流程：Wait, "identifying issues from scratch" might be part of a larger sentence like "rather than identifying issues from scratch." Since I only have this fragment, I'll translate it directly. In Chinese academic writing, fragments are sometimes kept as is if they follow a colon or dash, but here it's a period. I'll assume it's a continuation. I'll translate it as “从头识别问题”。“从头识别问题。元验证在供大语言模型掌握方面也更具样本效率。基于上述观察，我们开发了以下自动化标注流程：” 从头识别问题。元验证在供大语言模型掌握方面也具有更高的样本效率。基于上述观察，我们设计了以下自动化标注流程：“从头识别问题。大语言模型掌握元验证的样本效率也更高。基于上述观察，我们开发了以下自动化标注流程：” Actually, keeping the subject as "Meta-verification" is fine: “元验证在供大语言模型掌握方面也更具样本效率。” Both work. I'll stick with a balanced version. 从头识别问题。元验证在供大语言模型掌握方面也具有更高的样本效率。基于上述观察，我们设计了以下自动化标注流程：“从头识别问题。元验证在供大语言模型掌握方面也具有更高的样本效率。”

基于上述观察，我们设计了以下自动化标注流程：” Final: 从头识别问题。元验证在供大语言模型掌握方

面也具有更高的样本效率。基于上述观察，我们设计了以下自动化标注流程： Final decision: 从头识别问题。元验证在供大语言模型掌握方面也

2. For analyses reporting issues (scores 0 or 0.5), generate m meta-verification assessments

Actually, in Chinese academic writing, infinitive phrases at the start are often translated as purpose clauses: “以验证已识别的问题。” or “旨在验证已识别的问题。” I'll use “以验证已识别的问题。” For the second sentence: “若多数元评估证实其研究结果，则该分析被视为有效。” Combine: 以验证已识别的问题。若多数元评估证实其研究结果，则该分析被视为有效。 to validate the identified problems. -> 以验证已识别的问题。 An analysis is deemed valid if the majority of metaassessments confirm its findings -> 若多数元评估证实其研究结果，则该分析被视为有效。 Output: 以验证已识别的问题。若多数元评估证实其研究结果，则该分析被视为有效。“以验证已识别的问题。若多数元评估证实其研究结果，则该分析被视为有效。” matches academic standards perfectly. “以验证已识别的问题。若多数元评估证实其研究结果，则该分析被视为有效。” (Added slight academic polish if needed, but this is direct and accurate.) “以验证已识别的问题。若多数元评估证实其研究结果，则该分析被视为有效。” (Direct, academic, precise) [Output Generation] -> “以验证已识别的问题。若多数元评估证实其研究结果，则该分析被视为有效。” (I'll add a note about context if needed, but direct translation is preferred.)

3. For each proof, we examine analyses that assign the lowest score. If at least k such analyses

(若多个问题) 被认为有效，则该证明被标记为该最低分数。如果在所有验证尝试中均未发现有效问题，则该证明被标记为1。否则，该证明将被丢弃或转交人类专家进行标注。在最后的两次训练迭代中，这一全自动流水线完全取代了人工标注。质量检查证实，自动化标注结果与专家判断高度一致。若(问题) 被认定为有效，则该证明将被标记为该最低评分。若在所有验证尝试中均未发现有效问题，则该证明标记为1。否则，该证明将被丢弃或转交人类专家进行标注。在最后的两次训练迭代中，这一全自动流水线完全取代了人工标注。质量检查证实，自动化标注结果与专家判断高度一致。若(问题) 被认定为有效，则该证明将被标记为该最低评分。若在所有验证尝试中均未发现有效问题，则该证明标记为1。否则，该证明将被丢弃或转交人类专家进行标注。在最后的两次训练迭代中，这一全自动流水线完全取代了人工标注。质量检查证实，自动化标注

结果与专家判断高度一致。“若（相关缺陷）被认定为有效，则该证明将被赋予该最低评分。若在所有验证尝试中均未发现实质性错误，则该证明标记为1。否则，该证明将被剔除或转交人类专家进行标注。在最后的两次训练迭代中，该全自动流水线已完全取代人工标注。质量检查证实，自动化标注结果与专家判断高度吻合。”若（问题）被认定为有效，则该证明将被标记为该最低评分。若在所有验证尝试中均未发现有效问题，则该证明标记为1。否则，该证明将被剔除或转交人类专家进行标注。在最后的两次训练迭代中，这一全自动流水线完全取代了人工标注。质量检查证实，自动化标注结果与专家判断高度一致。

### 3. Experiments

我们采用组相对策略优化（Group Relative Policy Optimization, GRPO）（Shao 等，2024）进行强化学习，如第2节所述，迭代优化证明验证与生成能力。在每次迭代中，我们首先优化证明验证器。随后，证明生成器从验证器的检查点初始化，并针对证明生成任务进行优化。从第二次迭代开始，证明验证器通过拒绝微调（rejection fine-tuning）整合上一迭代中验证与生成能力的检查点进行初始化。内部CNML级别题目：共91道定理证明题，涵盖代数（13道）、几何（24道）、数论（19道）、组合数学（24道）和不等式（11道），难度与中国全国高中数学联赛（CNML）题目相当。

IMO 2025（6道题）：国际数学奥林匹克竞赛，面向全球中学生（大学前）的顶级数学竞赛。CMO 2024（6道题）：中国数学奥林匹克竞赛，中国国家级数学锦标赛。Putnam 2024（12道题）：威廉·洛厄尔·普特南数学竞赛，北美地区面向本科生的顶尖数学竞赛。

ISL 2024（31道题）：IMO短名单（IMO Shortlist），由参赛国提交、经题目遴选委员会审议并可能入选2024年IMO的题目集合。

IMO-ProofBench（60道题）：由DeepThink和IMO-Gold背后的DeepMind团队开发（Luong 和 Lockhart, 2025）。该基准（Luong 等，2025）分为基础集（30道题，难度从IMO前水平到IMO中等）和进阶集（30道高难度题目，模拟完整IMO考试，难度可达IMO困难级别）。我们首先评估模型在不进行迭代优化的情况下生成正确证明的能力。针对内部题目，我们为每个待评估模型每道题生成8个证明样本。证明的正确性通过最终验证器产生的8次验证分析结果进行多数投票判定。如图1所示，在所有CNML级别题目类别（代数、几何、数论、组合数学和不等式）中，DeepSeekMath-V2持续优于GPT-5-Thinking-High（OpenAI, 2025）和 Gemini 2.5-Pro（DeepMind, 2025）。

我们采用组相对策略优化（Group Relative Policy Optimization, GRPO）（Shao 等，2024）进行强化学习，如第2节所述，迭代优化证明验证与生成能力。在每次迭代中，我们首先优化证明验证器。随后，证明生

成器从验证器的检查点初始化，并针对证明生成任务进行优化。从第二次迭代开始，证明验证器使用一个检查点进行初始化，该检查点通过拒绝微调（rejection fine-tuning）融合了上一迭代中的验证与生成能力。

在多个领域展现出卓越的定理证明能力。对于来自IMO（国际数学奥林匹克）和CMO（中国数学奥林匹克）等竞赛的难题，模型在128K token限制内往往无法一次性生成全面且严谨的证明。当出现这种情况时，我们的证明生成器会通过自我验证识别出证明无效，但受限于上下文长度，无法在一次尝试中解决所有已发现的问题。为探索扩展上下文与自我验证如何提升证明质量，我们评估了结合自我验证的逐步优化方法。该方法首先生成带有自我分析的证明，随后利用其先前输出迭代式地重新提示生成器，使其能够针对已发现的问题进行修正。该过程将持续进行，直到生成器自我评分达到满分，或达到预设的最大顺序尝试次数为止。通过大规模验证来识别细微问题，并利用并行生成来探索多样化的证明策略。我们的方法为每个问题维护一个候选证明池，初始化为64个证明样本，并为每个样本生成64份验证分析。配对用于生成一个优化后的证明，随后更新候选证明池。该过程最多持续16次迭代，或直到某个证明成功通过全部64次验证尝试为止，这表明对其正确性具有高度置信度。

值得注意的是，对于未完全解决的题目，我们的生成器通常能够识别出其证明中的真实缺陷；而完全解决的题目则通过了全部64次验证尝试。这表明我们能够成功训练基于大语言模型的验证器，以评估以往被认为难以自动验证的证明。通过在验证器指导下扩展推理时计算（test-time compute），我们的模型能够解决那些需要人类选手耗费数小时才能攻克的题目。值得注意的是，对于未完全解决的题目，我们的生成器通常能准确识别其证明中的实质性错误；而完全解决的题目则全部通过了64次验证尝试。这表明，我们已成功训练出基于大语言模型的验证器，能够对以往被认为难以自动验证的证明进行有效评估。通过在验证器的指导下扩展推理时计算资源，我们的模型能够解决那些通常需要人类选手耗费数小时才能攻克的难题。值得注意的是，对于未完全解决的题目，我们的生成器通常能够准确定位其证明中的关键缺陷；而完全解决的题目则全部通过了64次验证尝试。这表明，我们已成功训练出基于大语言模型的验证器，能够对以往被认为难以自动验证的证明进行有效评估。通过在验证器的指导下扩展推理时计算（test-time compute），我们的模型能够解决那些通常需要人类选手耗费数小时才能攻克的难题。

### 4. Related Work

取得压倒性成绩。在一年内已在AIME和HMMT等定量推理基准测试中达到性能饱和。这一快速进步部分归因于其明确的评估标准：如果我们只关注最终答案，那么定量推理的结果很容易验证。然而，这种仅看最

终答案的评估指标并不适用于定理证明。定理证明通常不需要数值答案，而是要求严谨的逐步推导。非形式化数学证明长期以来一直被认为难以自动验证，缺乏可靠的方法来评估证明的正确性。近期的进展表明，这一障碍或许可以被克服。这构成了一个存在性证明，表明基于大语言模型 (LLM) 的复杂证明验证是可行的。证明必须使用形式化语言编写，但一旦通过编译，其正确性便得到保证。在 2024 年 IMO 中取得了银牌水平的成绩，但需要大量的计算资源。在 2025 年 IMO 中成功解出了 6 道题中的 5 道。取得近乎满分的成绩。

## 5. Conclusion

*\*Draft:* 我们提出了 DeepSeekMath-V2，一个能够生成和验证数学证明的模型。 *\*Refinement:*

本文提出了 DeepSeekMath-V2，该模型兼具数学证明生成与验证能力。 (More academic) *\*Draft:* 通过训练模型识别自身推理中的问题，并激励它们在最终输出前解决这些问题，我们超越了基于最终答案奖励的限制，走向可自我验证的数学推理。 *\*Refinement:* 通过训练模型识别自身推理过程中的缺陷，并激励其在生成最终输出前予以修正，我们突破了传统“基于最终答案奖励”机制的局限，迈向可自我验证的数学推理范式。

(More precise: "finalanswer-based rewards" -> 基于最终答案的奖励机制; "self-verifiable mathematical reasoning" -> 可自我验证的数学推理)

*\*Draft:* 我们的迭代训练过程——在提高验证能力和使用这些能力增强生成之间交替——创造了一个可持续的循环，其中每个部分都推动另一个前进。

*\*Refinement:* 我们的迭代训练流程在“提升验证能力”与“利用验证能力反哺生成质量”之间交替进行，构建了一个可持续的良性循环，使生成与验证两大模块相互促进、协同演进。 (More academic flow) *\*Draft:*

我们的关键技术贡献包括：(1) 训练一个准确且忠实的基于LLM的数学证明验证器，(2) 使用元验证大幅减少幻觉问题并确保验证质量，(3) 激励证明生成器通过自我验证最大化证明质量，以及 (4) 扩展验证计算以自动标记越来越难验证的证明，从而在无人标注的情况下改进验证器。 *\*Refinement:*

本文的核心技术贡献包括：(1) 训练出高精度且忠实可靠的基于大语言模型 (LLM) 的数学证明验证器；(2) 引入元验证 (meta-verification) 机制，大幅降低幻觉问题并保障验证质量；(3) 通过自我验证机制激励证明生成器最大化输出证明的质量；(4) 通过扩展验证计算资源，自动标注日益复杂的难验证证明，从而在无需人工标注的情况下持续优化验证器性能。 (Standardized terms: faithful -> 忠实可靠/高保真; meta-verification -> 元验证; hallucinated issues -> 幻觉问题/虚假问题; scaling verification compute -> 扩展验证计算资源/算力)

标题：DeepSeekMath-V2: Towards Self-Verifiable Mathematical Reasoning

作者：Zhihong Shao, Yuxiang Luo, Chengda Lu, Z.Z. Ren, Jiewen Hu, Tian Ye, Zhibin Gou, Shirong Ma, Xiaokang Zhang

提交日期：2025年11月27日

学科分类：cs.AI (主)，cs.CL

DOI：10.48550/arXiv.2511.22570

PDF 链接：<https://arxiv.org/pdf/2511.22570.pdf>

— AI 辅助翻译 · DeepSeek-Math-V2 · 仅供参考，请以英文原文为准 —