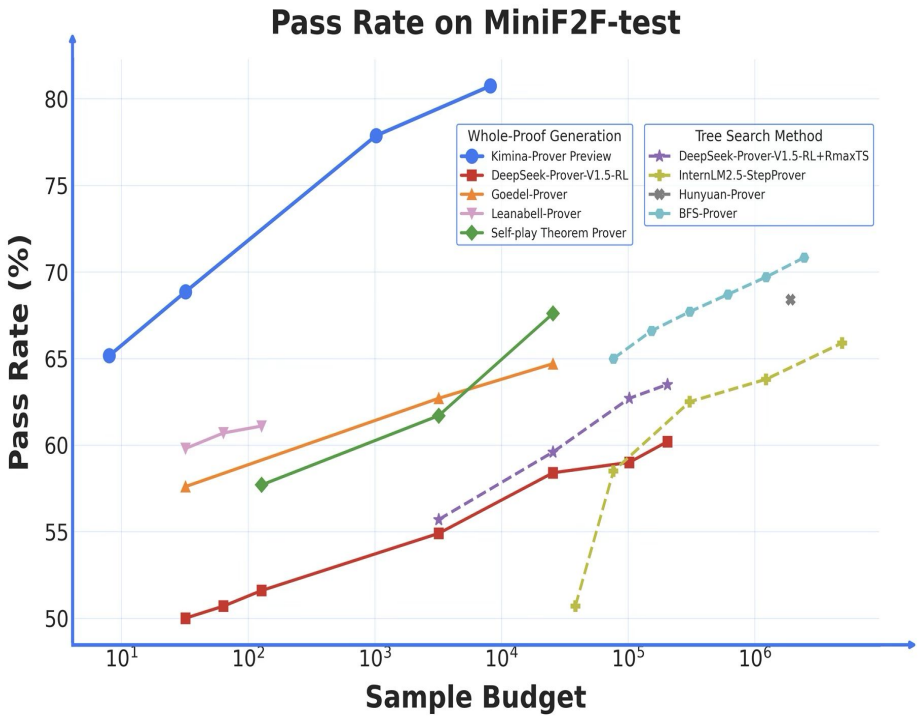
Kimina-Prover Preview

Towards Large Formal Reasoning Models with Reinforcement Learning

Numina & Kimi Team

Presenter: Jia LI (cofounder of Project Numina)



Problem

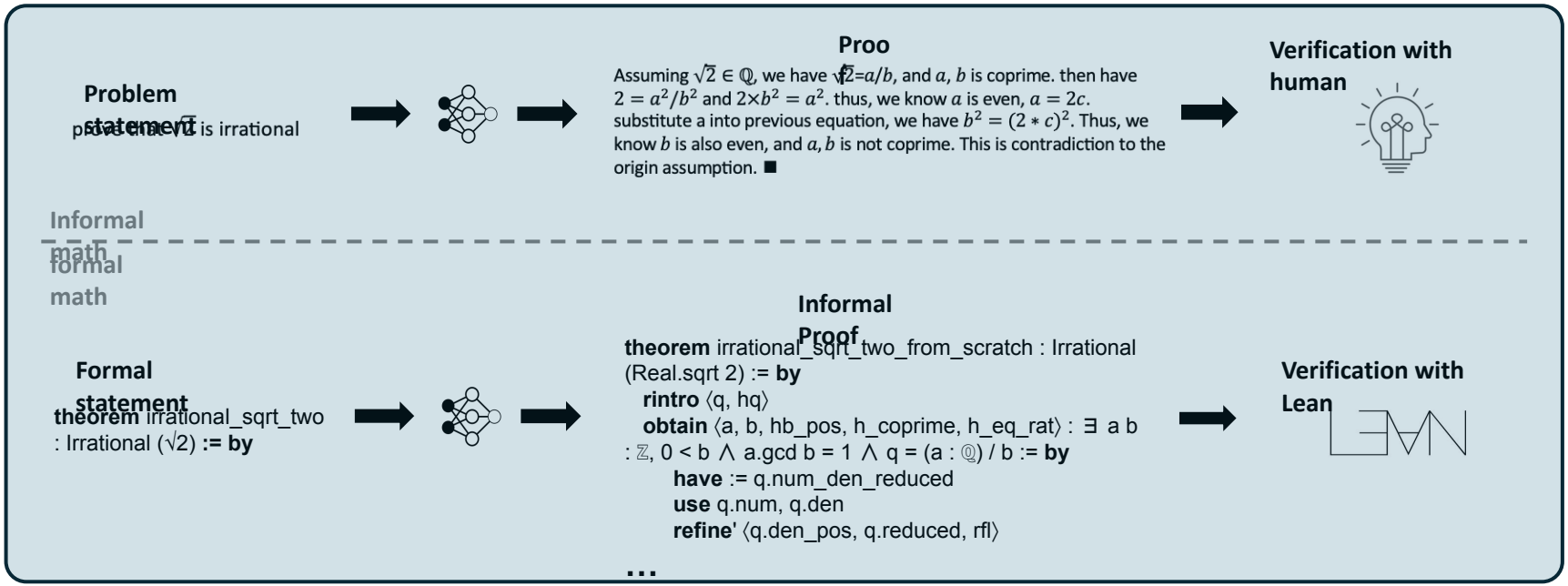
Automatic verifiable mathematical problem solving.

Pros:

- Automatically verifiable proofs, ensuring correctness.
- Interactive environments can show internal state.

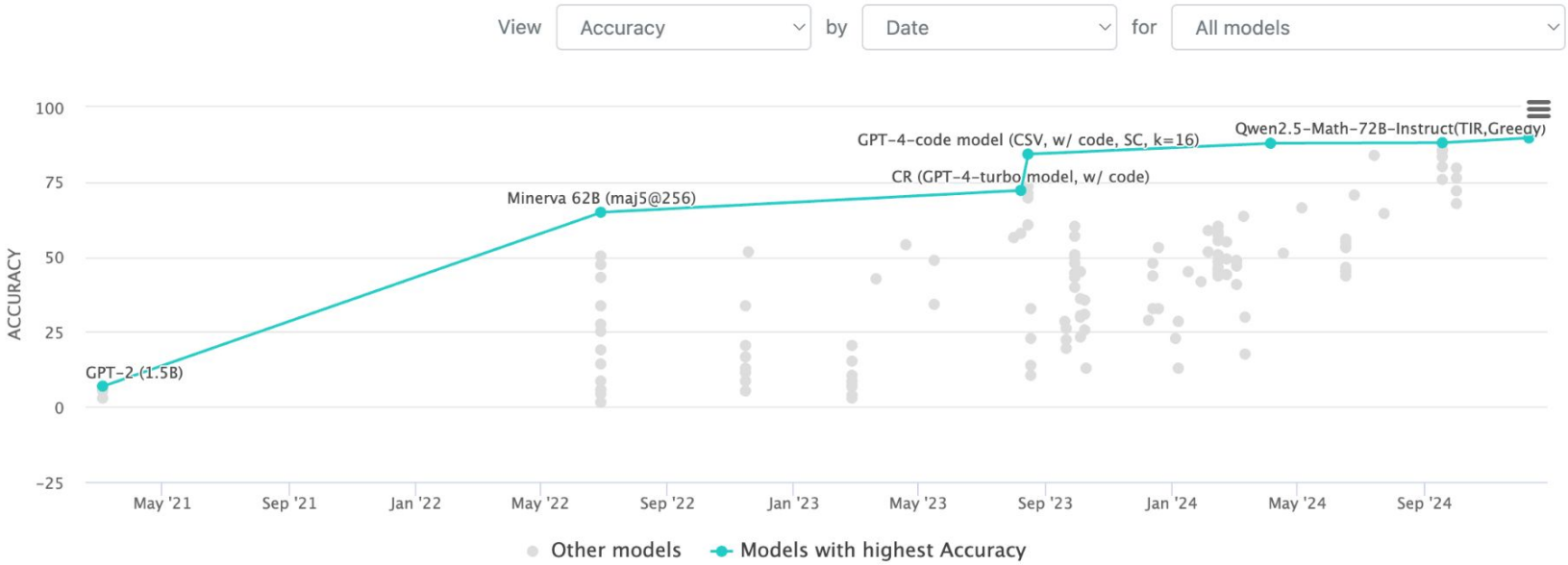
Cons:

- Data scarcity for training certain types of AI-based provers.
- Verbose in detailed steps

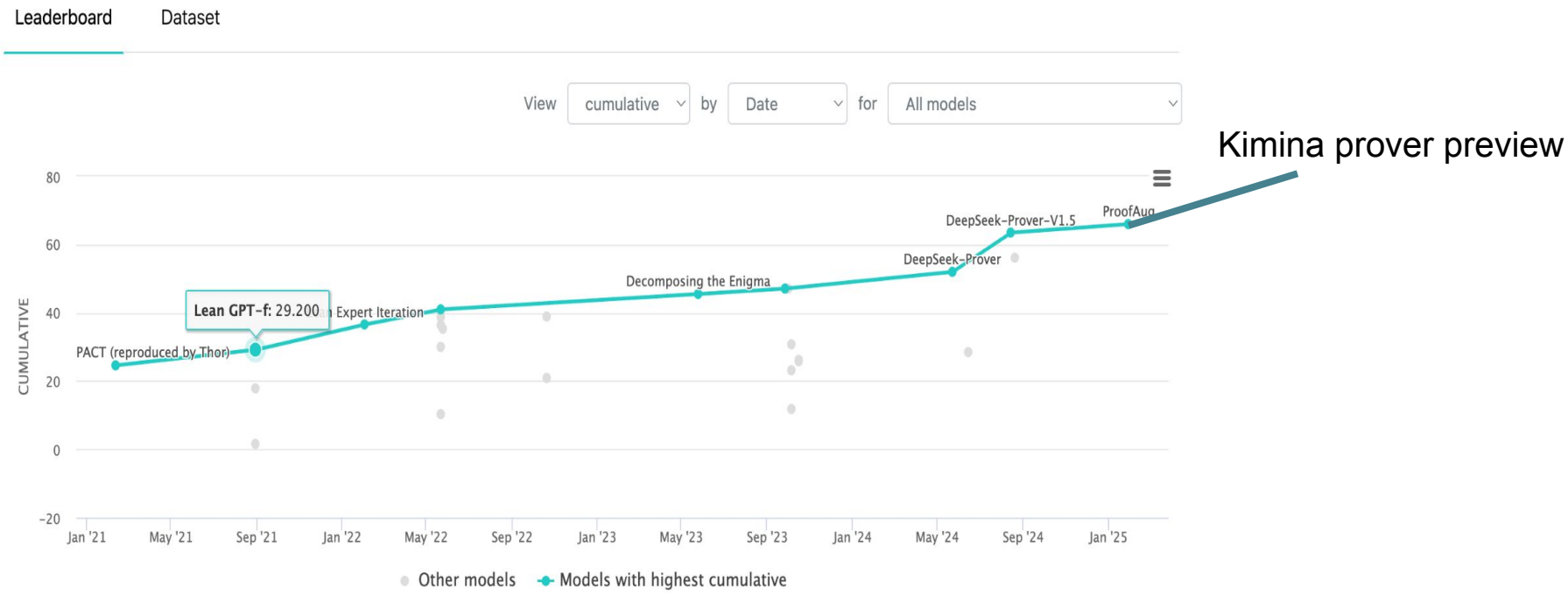


Math Word Problem Solving on MATH

Leaderboard Dataset



Automated Theorem Proving on miniF2F-test



Problem

Automatic verifiable mathematical problem solving.

Pros:

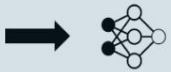
- Automatically verifiable proofs, ensuring correctness.
- Interactive environments can show internal state.

Cons:

- Data scarcity for training certain types of AI-based provers.
- Verbose in detailed steps

Problem statement

prove that $\sqrt{2}$ is irrational



Proof

Assuming $\sqrt{2} \in \mathbb{Q}$, we have $\sqrt{2}=a/b$, and a, b is coprime. then have $2 = a^2/b^2$ and $2 \times b^2 = a^2$. thus, we know a is even, $a = 2c$. substitute a into previous equation, we have $b^2 = (2 * c)^2$. Thus, we know b is also even, and a, b is not coprime. This is contradiction to the origin assumption. ■

Verification with human

Informal math

formal math

Formal statement

```
theorem irrational_sqrt_two :
Irrational (√2) := by
```

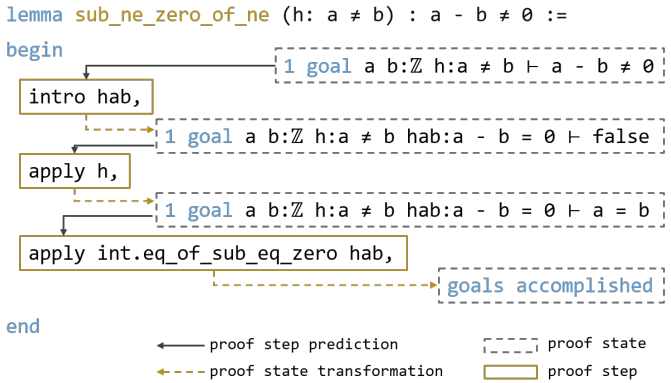
Informal Proof

```
theorem irrational_sqrt_two_from_scratch : Irrational
(Real.sqrt 2) := by
  rintro ⟨q, hq⟩
  obtain ⟨a, b, hb_pos, h_coprime, h_eq_rat⟩ : ∃ a b : ℤ, 0
< b ∧ a.gcd b = 1 ∧ q = (a : ℚ) / b := by
      have := q.num_den_reduced
      use q.num, q.den
      refine' ⟨q.den_pos, q.reduced, rfl⟩
...
```

Verification with Lean

Previous Approaches & Limitations

Step prover: Generating proof steps/tactics often coupled with classical search (BFS, MCTS).



Whole proof: Generating whole proof structures at once.

```
theorem irrational_sqrt_two_from_scratch :
Irrational (Real.sqrt 2) := by
```

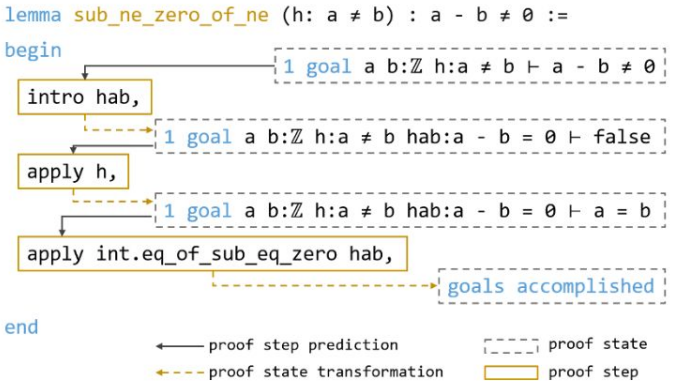
```
  rintro ⟨q, hq⟩
  obtain ⟨a, b, hb_pos, h_coprime, h_eq_rat⟩ : ∃ a b : ℤ,
0 < b ∧ a.gcd b = 1 ∧ q = (a : ℚ) / b := by
    have := q.num_den_reduced
    use q.num, q.den
    refine' ⟨q.den_pos, q.reduced, rfl⟩
…
```

Limitations:

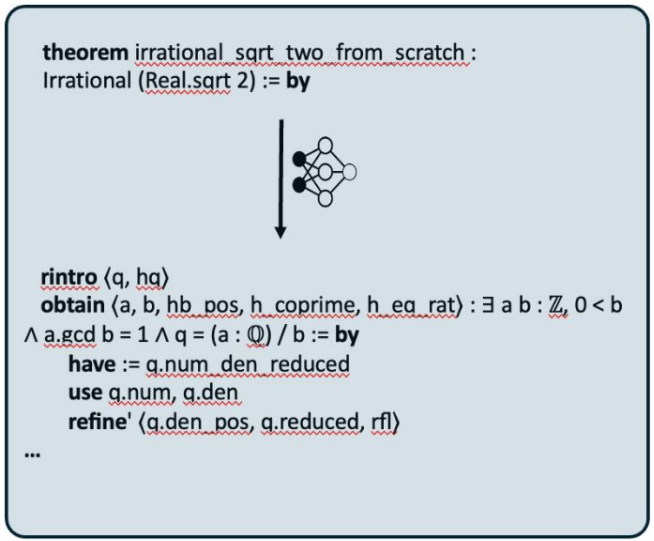
- Step prover are performant but reliance on explicit search is computationally expensive and complex
- Whole proof generation are swift but LLMs struggle with deep, non-linear formal reasoning
- **Lack of clear performance scaling with model size.**

Previous Approaches & Limitations

Step prover: Generating proof steps/tactics often coupled with classical search (BFS, MCTS).



Whole proof: Generating whole proof structures at once.



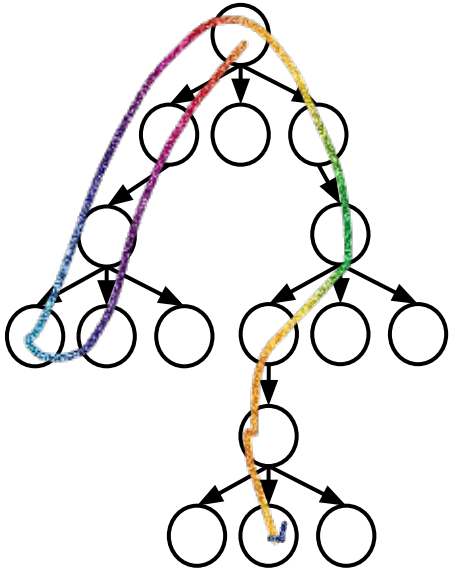
Limitations:

- Step prover are performant but reliance on explicit search is computationally expensive and complex
- Whole proof generation are swift but LLMs struggle with deep, non-linear formal reasoning
- **Lack of clear performance scaling with model size.**

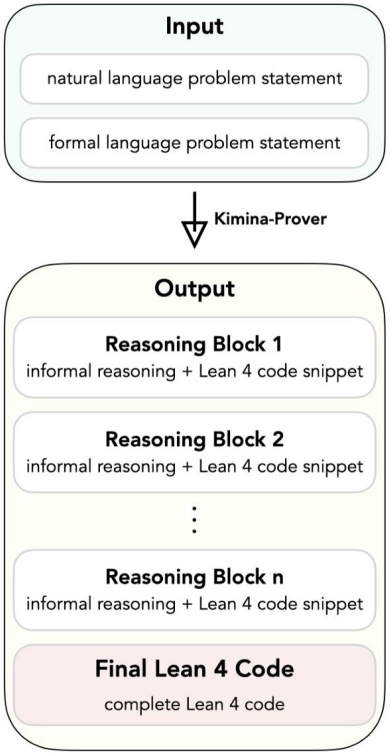
Key Idea: Formal Reasoning Models

Bring O1-style reasoning into formal mathematics:

- Keep step prover's performant with reasoning cot
- Keep whole proof model's swift with single pass for each problem



Flatten proof search trajectories into single reasoning COT



Method Overview

- **Autoformalization for Data:**
  - Need large, diverse Lean 4 problems for RL training.
  - Trained models (like Kimina-Autoformalizer-7B) to translate natural language math problems into formal Lean 4 statements. Overcame lack of direct reward signal using SFT and LLM judging.

- **Formal Reasoning Pattern:**
  - Designed to align informal mathematical thinking with formal Lean 4 code generation.
  - Uses special tags (<think>, code markers) to intersperse informal reasoning and formal snippets.

- **Reinforcement Learning:**
  - Built on Kimi k1.5 RL pipeline.
  - Refines the model's ability to generate correct proofs using the formal reasoning pattern.

Formal Reasoning Pattern

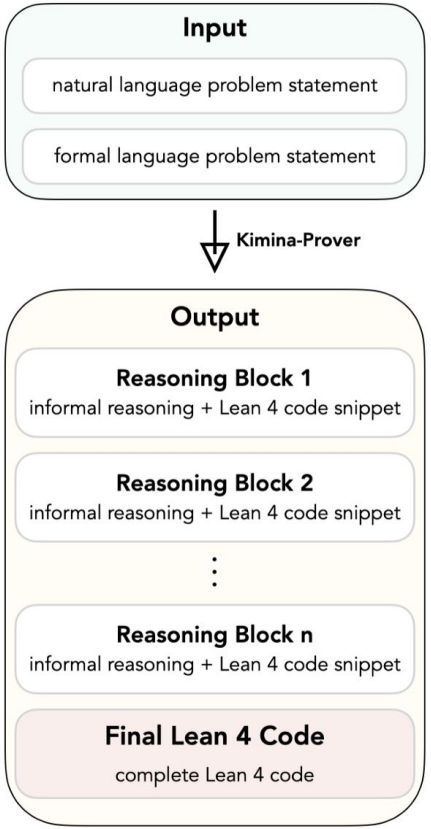
Structure

- Input: Formal problem statement.
- Output: Sequence of reasoning blocks (<think>...</think>) containing informal thoughts + Lean snippets, followed by final complete Lean code.

Goal: Align informal intuition and formal steps. Ensure thinking steps correspond to final proof.
Benefits: Improves explainability (can inspect the model's "thought process"). Correlates with proof complexity.

Training Boost:

- ***Cold Start****:* Initial SFT using ~20k examples synthesized by Claude 3.7 Sonnet to teach the pattern.
- ***Informal Math Mix-Training:*** Incorporated informal math thinking data to improve reflection quality during RL

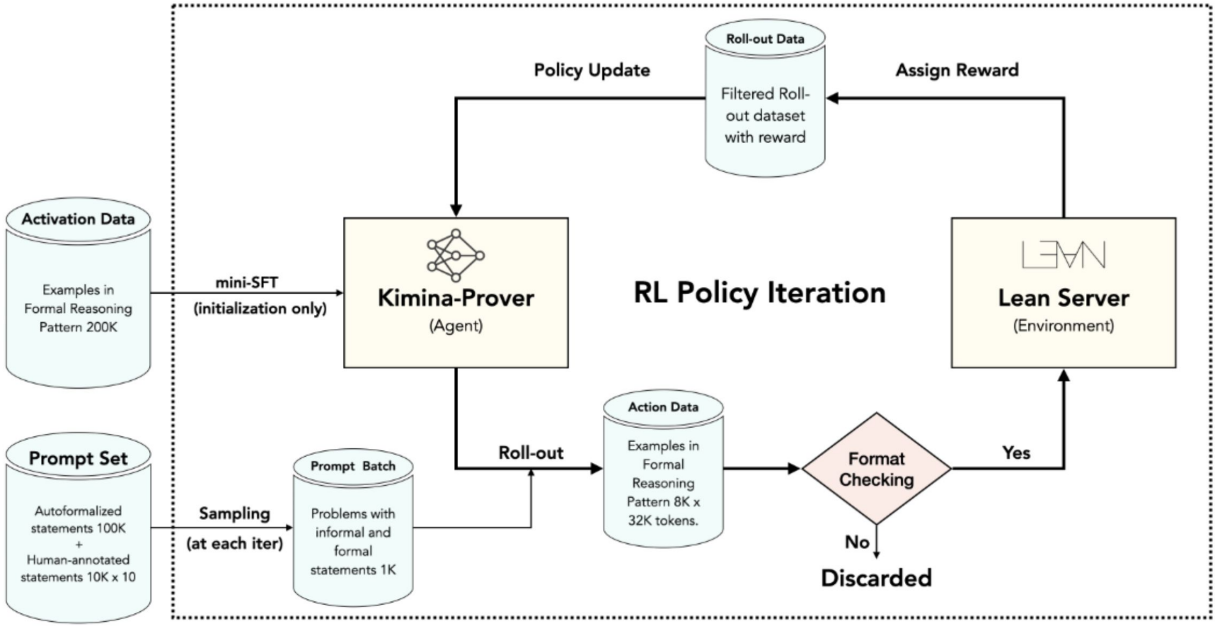


Reinforcement Learning

Bring Test Time Scaling to Formal Reasoning Pattern

To prevent "format collapse" during RL.

- Filter rollouts to ensure they follow the reasoning pattern (contain tactic blocks, cover final code)
- Randomly discard some negative gradient samples.



Results

Prover system	Model size	Sample budget	miniF2F-test
Tree search systems			
DeepSeek-Prover-V1.5-RL + RMaxTS (H. Xin et al. 2024)	7B	$32 \times 16 \times 400$	63.5%
InternLM2.5-StepProver-BF+CG (Wu et al. 2024)	7B	$256 \times 32 \times 600$	65.9%
HunyuanProver v16+BFS+DC (Y. Li et al. 2025)	7B	$600 \times 8 \times 400$	68.4%
BFS-Prover (R. Xin et al. 2025)	7B	$2048 \times 2 \times 600$	70.8%
Whole-proof systems			
DeepSeek-Prover-V1.5-RL (H. Xin et al. 2024)	7B	102400	60.2%
Goedel-Prover-SFT (Y. Lin et al. 2025)	7B	25600	64.7%
Leanabell-Prover (Zhang et al. 2025)	7B	128	61.1%
Kimina-Prover-Preview-Distill-1.5B	1.5B	1	42.6%
		32	56.2%
		1024	**61.9%**
Kimina-Prover-Preview-Distill-7B	7B	1	52.5%
		32	63.1%
		1024	**70.8%**
Kimina-Prover-Preview	72B	1	52.94%
		8	65.16%
		32	68.85%
		1024	77.87%
		8192	**80.74%**

Benchmark: miniF2F, a standard test for formal math with diverse difficulty level

State-of-the-Art: Kimina-Prover Preview achieves **80.74%** pass@8192, significantly outperforming previous methods (including tree search).

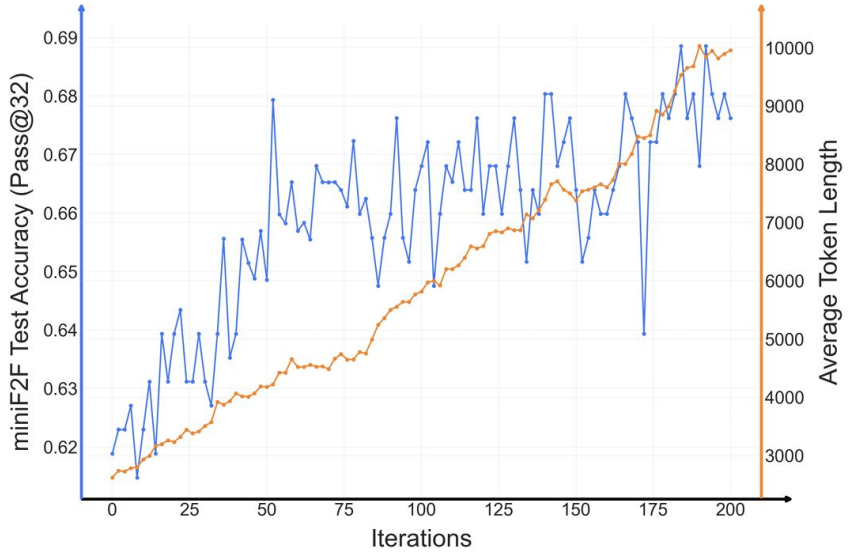
High Sample Efficiency:

- Strong performance even with few samples: **52.94% pass@1**.
- Reaches **68.85% pass@32**, competitive with prior SOTA needing far more samples.

Key Findings: Test Time Scaling

Test Time Scaling: Performance (pass@32 accuracy) improves as the model learns to generate longer, more complex outputs during training

- Formal reasoning scaling is more volatile than informal math but ultimately successful. Suggests approach may work in other data-limited domains.



Key Findings: Human-like Proof Style

Emergent Human-like Style: The model learns complex reasoning patterns beyond simple step generation:

- Exploring multiple paths, reflection, refining thinking.
- Generating structured, decomposed proofs (e.g., using have statements like humans).

```
theorem imo_1962_p2 (x : ℝ) (h₀ : 0 ≤ 3 - x) (h₁ : 0 ≤ x + 1)
    (h₂ : 1 / 2 < Real.sqrt (3 - x) - Real.sqrt (x + 1)) : -1 ≤ x ∧ x < 1 - Real.sqrt 31 / 8 := by
  have h3 : -1 ≤ x := by
    nlinarith
  have h4 : Real.sqrt (3 - x) - Real.sqrt (x + 1) > 1 / 2 := by linarith
  have h5 : Real.sqrt (x + 1) ≥ 0 := Real.sqrt_nonneg (x + 1)
  have h6 : (7 / 4 - 2 * x) > Real.sqrt (x + 1) := by
    nlinarith [Real.sq_sqrt (show (0 : ℝ) ≤ 3 - x by linarith), Real.sq_sqrt (show (0 : ℝ) ≤ x +
    1 by linarith),
      Real.sqrt_nonneg (3 - x), Real.sqrt_nonneg (x + 1)]
  have h7 : (7 / 4 - 2 * x) ^ 2 > (Real.sqrt (x + 1)) ^ 2 := by
    nlinarith [h6, Real.sqrt_nonneg (x + 1)]
  have h8 : (7 / 4 - 2 * x) ^ 2 > x + 1 := by
    have h10 : (Real.sqrt (x + 1)) ^ 2 = x + 1 := by
      rw [Real.sq_sqrt]
      linarith
    nlinarith [h7, h10]
  have h9 : 64 * x ^ 2 - 128 * x + 33 > 0 := by
    nlinarith [h8]
  have h10 : x < 1 - Real.sqrt 31 / 8 := by
    by_contra hx
    push_neg at hx
    have h12 : Real.sqrt 31 > 0 := by
      apply Real.sqrt_pos.mpr
      norm_num
    nlinarith [sq_nonneg (x - (1 - Real.sqrt 31 / 8)), sq_nonneg (Real.sqrt 31), Real.sq_sqrt
    (show (0 : ℝ) ≤ 31 by norm_num),
      h12]
  exact ⟨h3, h10⟩
```

Listing 3: Lean 4 proof of IMO-1962-P2 found by `Kimina-Prover`.

```
theorem imo_1962_p2 (x : R)
  (h0 : 0 <= 3 - x)
  (h1 : 0 <= x + 1)
  (h2 : 1/2 < sqrt(3 - x) - sqrt(x + 1)) :
  -1 <= x and x < 1 - sqrt(31)/8 := by {
    constructor
    linarith
    rw [← sub_pos]
    field_simp [Real.sqrt_lt] at h2 |-
    apply lt_of_le_of_lt
    rw [mul_comm]
    rw [sub_eq_add_neg]
    apply lt_of_le_of_lt
    rw [← lt_sub_iff_add_lt]
    ring_nf
    rw [← lt_sub_iff_add_lt]
    linarith [Real.sq_sqrt (by linarith : 0 <= 1 + x)]
    rw [Real.sqrt_lt (by norm_num)]
    rw [Real.sqrt_lt] <;> nlinarith
    norm_num at this
  }
```

Listing 4: Lean 4 proof of IMO-1962-P2 found by `BFS-Prover`.

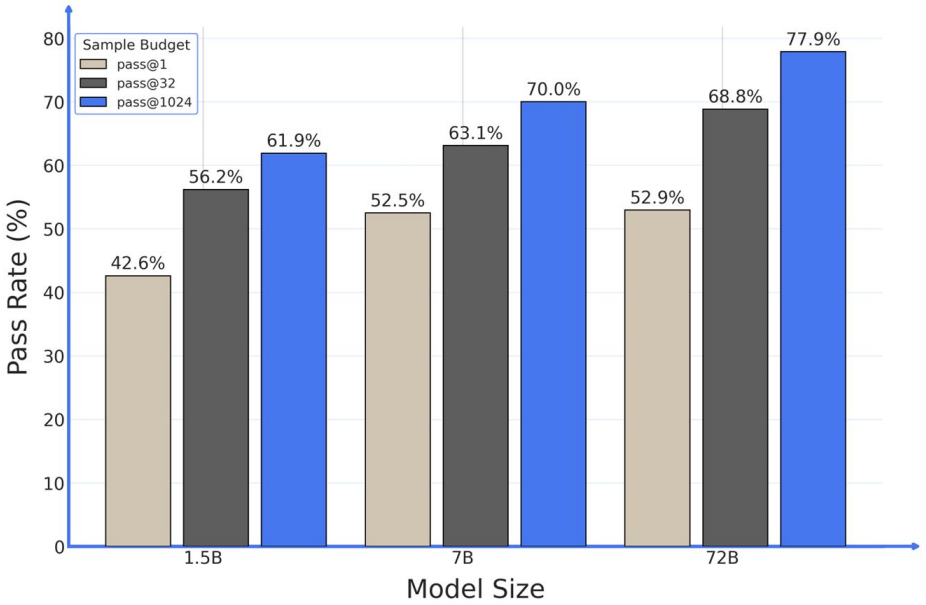
Key Findings: Scaling with Model Size

Observation: Clear performance improvement as model size increases (1.5B -> 7B -> 72B parameters).

- 72B model significantly outperforms 7B, especially with larger sample budgets (+7.87% at pass@1024).
- This scaling trend ***was not clearly observed*** in previous neural theorem provers for formal math.

Possible Cause:

- The use of informal math reasoning COT
- Flattened Reasoning path for Formal mathematics



Conclusion & Future Work

Kimina-Prover Preview demonstrates a promising new direction for neural theorem proving.

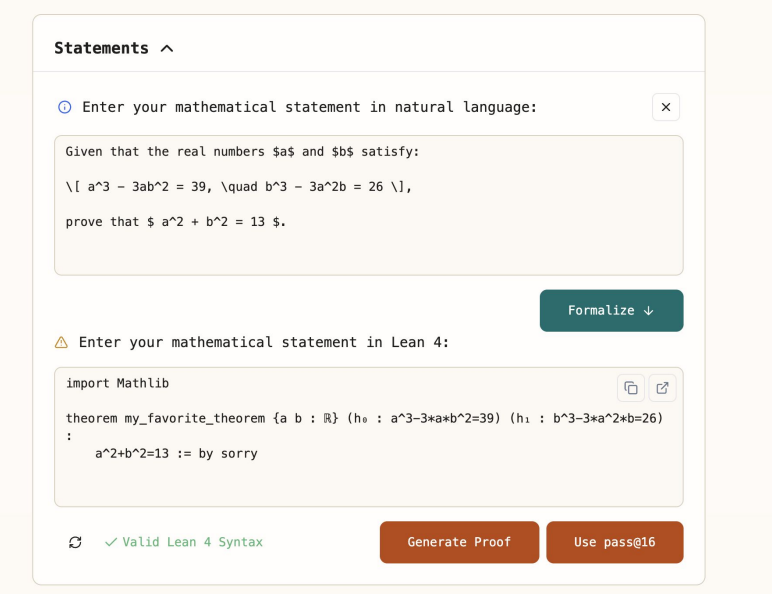
It's just a start, there are many more to be done! Reasoning model is extremely flexible and capable of incorporating most of the idea in the field of neural theorem proving!

Future Directions:

- Improve proof quality (e.g., filter outputs relying too heavily on high-level tactics).
- Enable iterative refinement using Lean compiler feedback for error correction.
- Integrate external tools (library search, computation engines).

Kimina demo is available

- https://github.com/MoonshotAI/Kimina-Prover-Preview
- Distill 1.5B model and 7B are available
- Demo of the 72B model
- Dataset and the 72B model will be made available soon



Future of Formal Theorem proving

1. High school level competition math will be completely solved by the end of 2025. LLM will be able to create IMO level problems and train student.
2. LLM become a necessary tool for mathematician as part of their research
3. Large scale autoformalization of existing theorem and papers -> “one step closer to game over”
4. LLM can help human to work on open problems