AI for Mathematical Discovery:

Symbolic, Neural and Neuro-Symbolic Methods

Moa Johansson

EuroProofNet Summer School 2025, Bonn

Mathematical Discovery and AI

- *“Within ten years a digital computer will discover and prove an important new mathematical theorem”*

(Newell & Simon 1958)

- *“I expect, say, 2026-level AI, when used properly, will be a trustworthy co-author in mathematical research, and in many other fields as well. Strangely, even nonsensical LLM-generated math often references relevant concepts.”*

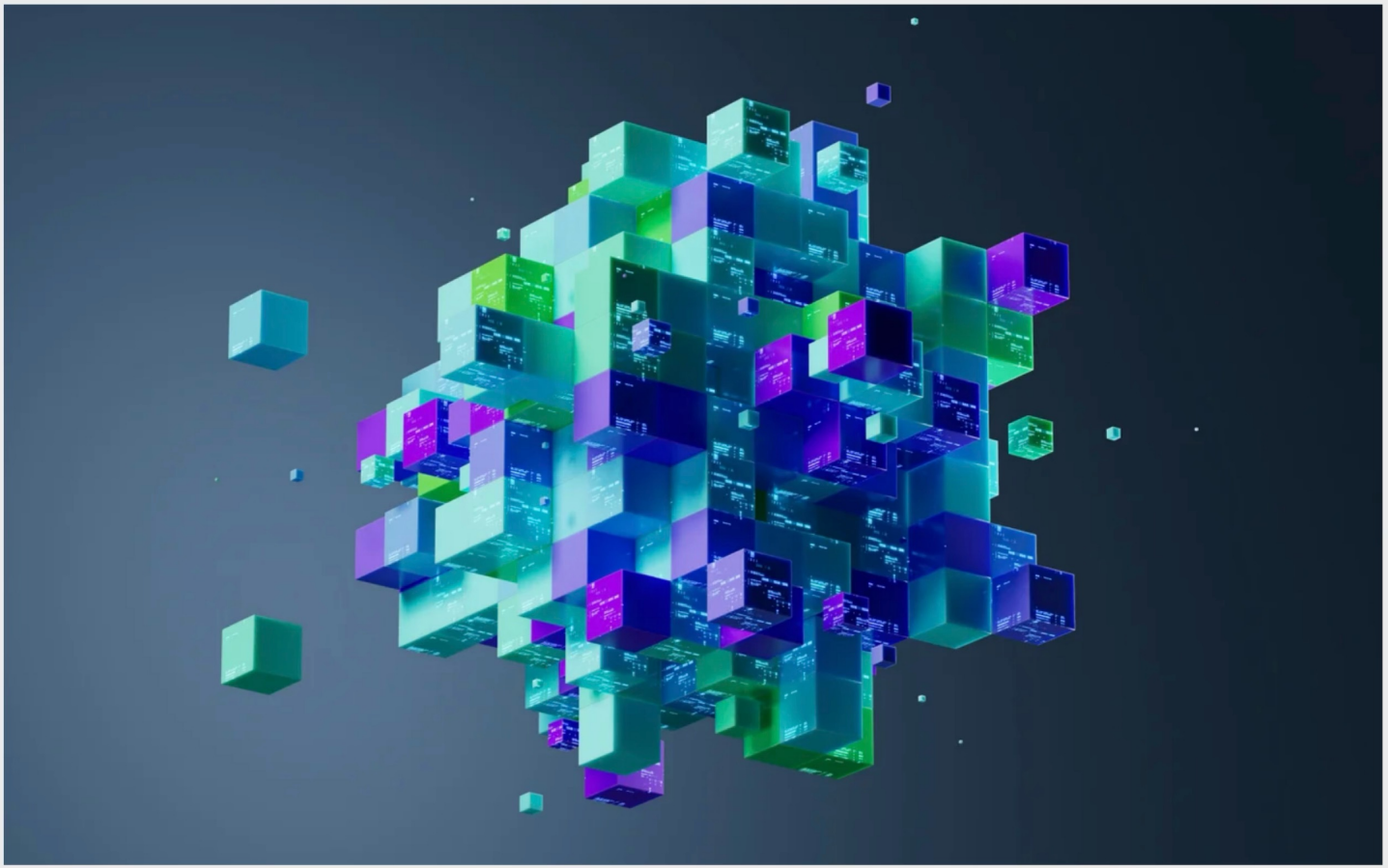
(Terence Tao 2023)

JUNE 8, 2024 | 12 MIN READ

AI Will Become Mathematicians' 'Co-Pilot'

Fields Medalist Terence Tao explains how proof checkers and AI programs are dramatically changing mathematics

BY CHRISTOPH DRÖSSER



UNIVERSITY OF GOTHENBURG

CHALMERS UNIVERSITY OF TECHNOLOGY

Theory Exploration aka Conjecturing

One piece of the puzzle?

- Assist invention of conjectures about mathematical theories.
  - New, interesting and non-trivial.
- Three approaches:
  - **Symbolic:** classic AI methods – heuristic search using grammars, rules etc.
  - **Neural:** modern machine learning based methods, often using Large Language Models (LLMs) or other generative neural networks.
  - **Neuro-symbolic:** combination of the two – can we get the best of both worlds?

Symbolic Methods

Symbolic Methods

- Long history of heuristic/search-based methods:
  - **AM** (Lenat 1976),
  - **Grafitti** (Fajtlowicz 1988),
  - **HR(L)** (Colton/Pease 2000s),
  - **MATHsAiD** (McCasland 2010),
  - **IsaCoSy** (Johannson et al 2011),
  - **IsaScheme** (Montano-Rivas et al. 2012),
  - **TheoryMine** (Bundy et al 2015).
  - **QuickSpec 2** (Smallbone et al. 2017).
- Application example: Generate lemmas for automated provers
  - e.g. *Lemma Discovery and Strategies for Induction* (Einarsdottir et al. IJCAR 2024).

Symbolic conjecturing: QuickSpec

The Octonions

- Least known of the four normed division algebras:
  - reals, complex numbers and quaternions.
- Definitions in Haskell program (or TIP, a SMTLIB-like format).
- No proofs - conjecture suggestions via automated testing of terms in equivalence classes.

Quick specifications for the busy programmer. N. Smallbone, M. Johansson, K. Claesson, M. Algehed. Journal of Functional Programming 2017.

Data-driven methods

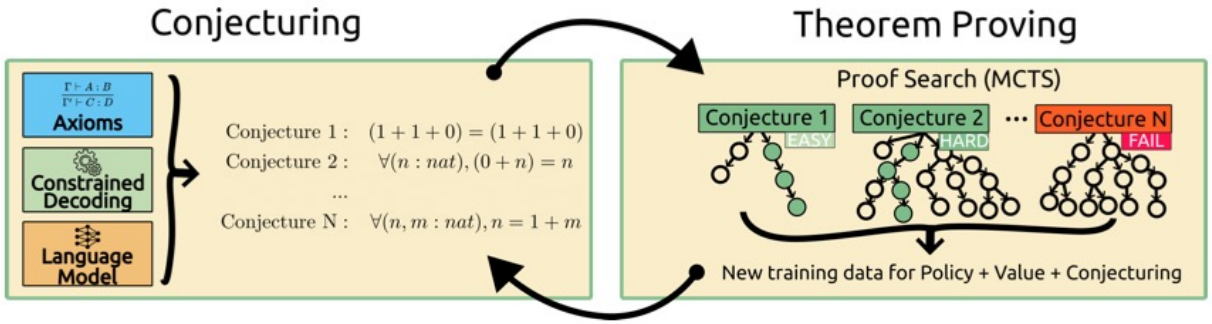
Still (mostly) symbolic

- **Idea:** discovery by analogy to other lemmas.
- Templates (skeleton lemmas with “holes”) extracted, then synthesise instantiations for holes.
  - Proof pattern recognition for ACL2 (Heras et al. 2013). Symbols encoded as numeric vectors. Look for analogies in this space.
  - RoughSpec (Einarsdottir et al. 2021): make QuickSpec more efficient by restricting search space to certain shapes.
- Can templates be learned?

Neural Methods

Neural Methods

- GPT-2 model trained on Mizar (Urban & Jakubův 2020)
  - Generated new conjectures when temperature set just right.
- Conjecturing for HOL Light (Rabe et al. 2021)
  - Skip-tree architecture. 10-30% new & interesting, rest false or repetitions.
- MINIMO – RL to generate novel conjectures (Poesia et al, NeurIPS 2024)
  - Very restricted domains: propositional logic, Peano arithmetic, group theory.
  - “Game” of conjecturing and proofs starting from the axioms.
  - Neural model using *constrained decoding* to produce well-formed conjectures.



Alpha Geometry

- **AlphaGeometry** – domain specific conjectures (Trinh et al, Nature 2024)
  - Very performant on International Math Olympiad problems in geometry, including learning to suggest additional constructs (lines, points etc) in Euclidian plane geometry.

SCIENCE

AI achieves silver-medal standard solving International Mathematical Olympiad problems

25 JULY 2024

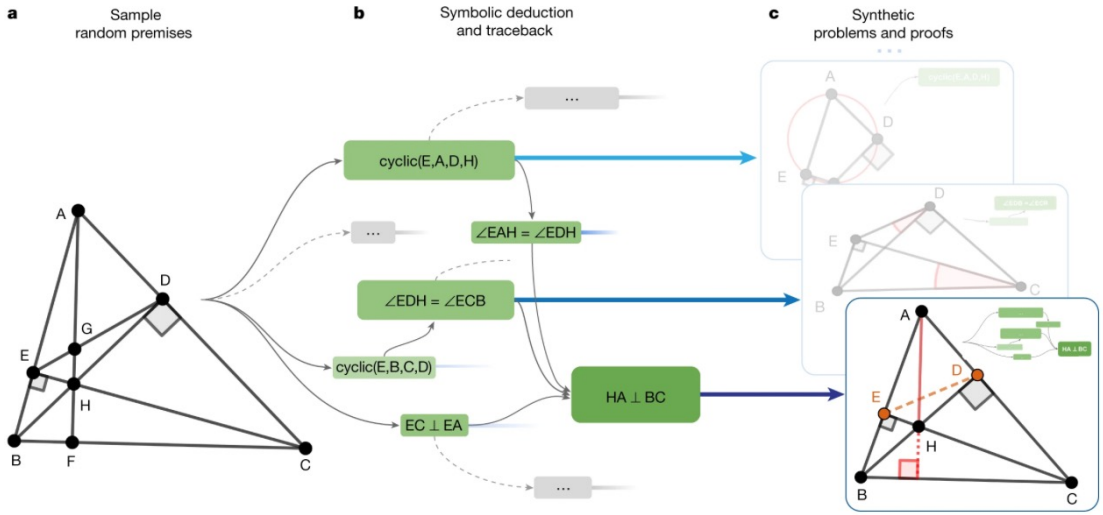
AlphaProof and AlphaGeometry teams

AlphaGeometry

Generating Examples with Auxiliary Constructs

Fig. 3: AlphaGeometry synthetic-data-generation process.

From: Solving olympiad geometry without human demonstrations



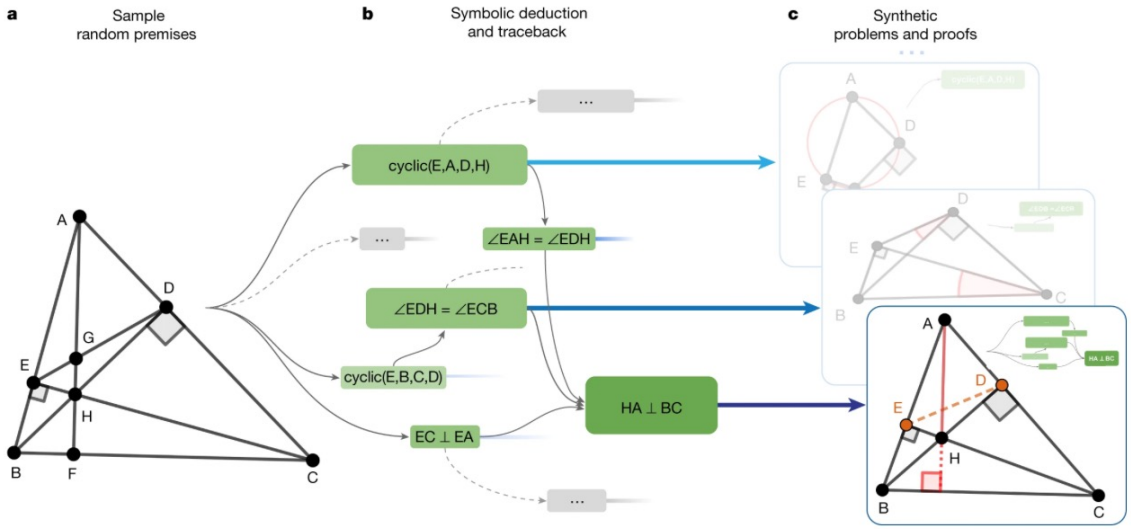
Alpha Geometry

Generating Examples with Auxiliary Constructs

- Synthetic training example:
  - (premises, conclusion, proof) = (P, N, G(N))
- Candidate “auxiliary point” which we want to train LLM to discover:
  - *Subset of P not appearing in N.*
- Example:
  - N: $HA \perp BC$ has in its derivation points E, D.
  - But these do not appear in N itself.
  - Remove E, D from premises, add as extra discovery steps in proof instead.
- This results in a training data point with a discovery step!

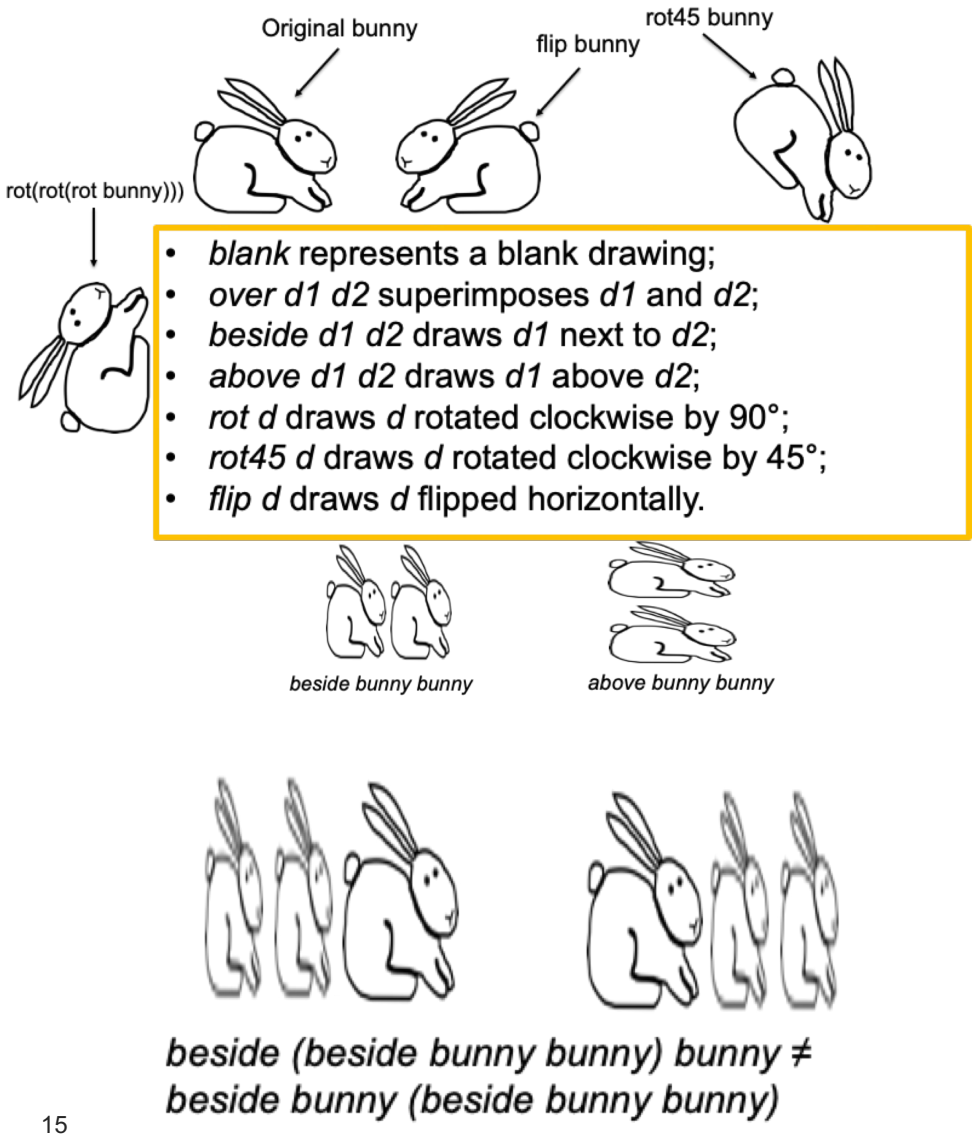
Fig. 3: AlphaGeometry synthetic-data-generation process.

From: Solving olympiad geometry without human demonstrations



AlphaGeometry: Data

- How was the neural part trained to suggest new points?
  - Number of human-written proofs is limited.
- Generated **100 million** synthetic data examples:
  - Generation of **1 billion** random diagrams of geometric objects.
  - Derived all relationships between points and lines in diagram.
  - Symbolic part searched for all proofs contained in each diagram (reasoning forwards).
  - **In each proof, check for steps with "additional constructs" appearing in the middle of reasoning chain. These are e.g. points not in the initial and final states.**
  - Filtered to remove similarities and duplicates.
  - **Nine million featured "additional constructs".**
- This is enough to train an LLM from scratch!



Using a pre-trained LLM instead?

- What if we don't have millions of synthetic datapoints?
- Use a LLM to generate lemmas zero shot for our favourite proof assistant?
- Common math theories will be in training data.
  - New formalisation in Isabelle/HOL.
- Caveat: This benchmark is however online as part of the QuickSpec benchmarks in Haskell.

Here are some lemmas about the functions blank, over, beside, above, rot, flip, and rot45 in Isabelle format. Note that these lemmas have not been proven and use 'sorry' as a placeholder for proofs.

arduino

```
theory Draw2Lemmas
  imports Draw2
begin

lemma blank_over_left: "over blank d = d"
  sorry

lemma blank_over_right: "over d blank = d"
  sorry

lemma over_associative: "over (over d1 d2) d3 = over d1 (over d2 d3)"
  sorry

lemma over_commute: "over d1 d2 = over d2 d1"
  sorry

lemma beside_blank_left: "beside blank d = d"
  sorry

lemma beside_blank_right: "beside d blank = d"
  sorry

lemma beside_associative: "beside (beside d1 d2) d3 = beside d1 (beside d2 d3)"
  sorry

lemma beside_commute: "beside d1 d2 = beside d2 d1"
  sorry

lemma above_blank_top: "above blank d = d"
  sorry

lemma above_blank_bottom: "above d blank = d"
  sorry

lemma above_associative: "above (above d1 d2) d3 = above d1 (above d2 d3)"
  sorry

lemma above_commute: "above d1 d2 = above d2 d1"
  sorry

lemma rot_rot: "rot (rot d) = d"
  sorry

lemma flip_flip: "flip (flip d) = d"
  sorry

lemma rot45_rot45_rot90: "rot45 (rot45 d) = rot d"
  sorry

lemma rot45_flip_rot45: "rot45 (flip (rot45 d)) = d"
  sorry

end
```

How did GPT-4 do?

- Many conjectures were false.
  - Not huge problem - checked by theorem prover/counter example checker.
- Appear to have internalised some “templates”
  - There is an **identity element**.
  - **Binary** functions are **associative and commutative**.
  - **Unary** functions are their **own identity**.
  - Binary functions **distribute over one another** (sometimes).
- Misses some lemmas symbolic system finds:
  - over x x = x
  - Equivalent 2 x 2 grid layouts:
    - above (beside x y) (beside z w) = beside (above x z) (above y w)

Exploring Mathematical Conjecturing with Large Language Models. Moa Johansson, Nicholas Smallbone. NeSy 2023, 17th International Workshop on Neural-Symbolic Learning and Reasoning

Symbolic vs. Neural Conjecturing

UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY

- **Less restricted shapes** of lemmas in neural systems.
  - QuickSpec it tailored for equalities.
  - QuickSpec fails to find conjectures outside its size limit (but given more compute it would).
  - rot45 (rot45 (rot45 (rot45 (rot45 (rot45 (rot45 (rot45 x)))))))) = x
- LLMs can use **information from function names**
  - Rotation lemmas - rotate correct number of times (not always though!)
  - Suggestions of “extra” auxiliary functions to include.
- Buggy definitions?
  - QuickSpec will **miss or discover other properties**.
  - **LLM may still suggest “intended” property**, even though there is a bug in the function definition.
  - *Seeking Specifications: The Case for Neuro-Symbolic Specification Synthesis*. George Granberry, Wolfgang Ahrendt, Moa Johansson. Forthcoming in Journal of Symbolic Computation 2025. Preprint: https://arxiv.org/abs/2504.21061

Neuro-Symbolic Methods

Discovering Lemmas by Analogy

- **Can we leverage analogies between mathematical domains to suggest conjectures?**
- Recall: Symbolic work on **templates** capturing common lemma patterns.
  - Can we learn templates from data?
  - But instantiate them symbolically?

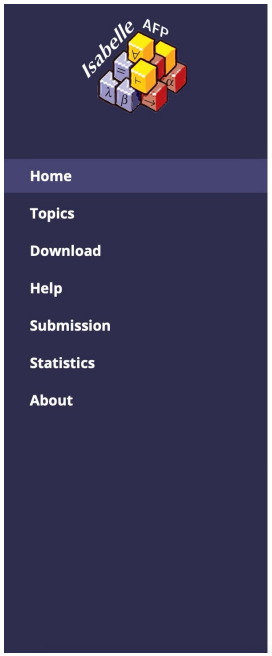
- Method should be general: applicable across **different mathematical domains**.
- Suggest lemmas in a **proof assistant**: speed up formalisations.
  - Proof assistant provides counter-example checking, tactics for proofs.

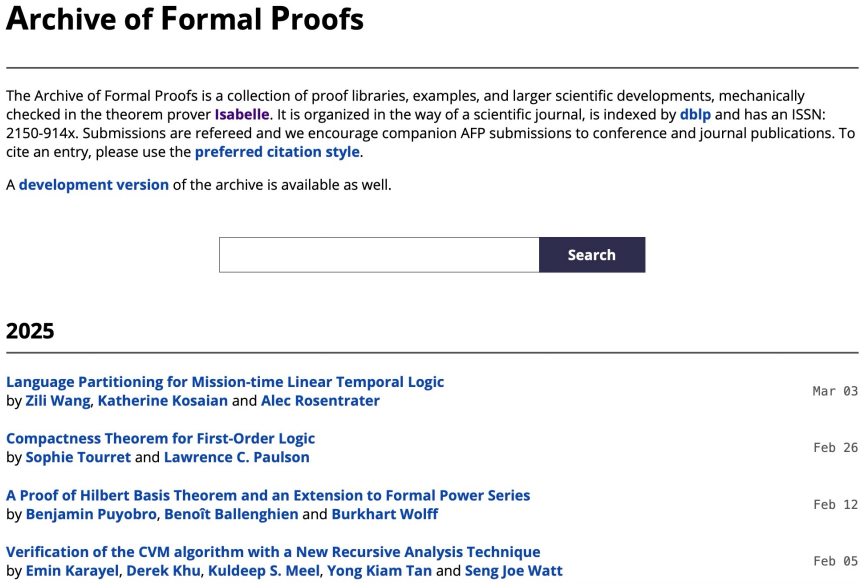
Archive of Formal Proofs

UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY

Isabelle Proof Assistant

- Formalisation in
  - Computer Science
  - Logic
  - Mathematics
- 890 entries
  - ~284,000 lemmas
- Can we discover this type of lemmas?





https://www.isa-afp.org/

Analogies in Isabelle's AFP

- Observation: Statements in proof libraries often **share some structure**.
- Extract from Isabelle's AFP - abstract to get templates.
- Smallish number of these are much more common (Einarsdottir AITP 2022).
- Can we **train a neural model to suggest which analogies to make** to a new theory?
  - i.e. which templates to suggest.
- **Generate conjectures** directly or **instantiate templates symbolically**?

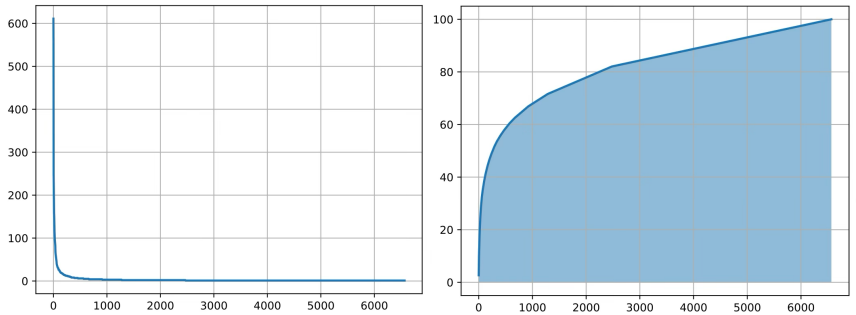
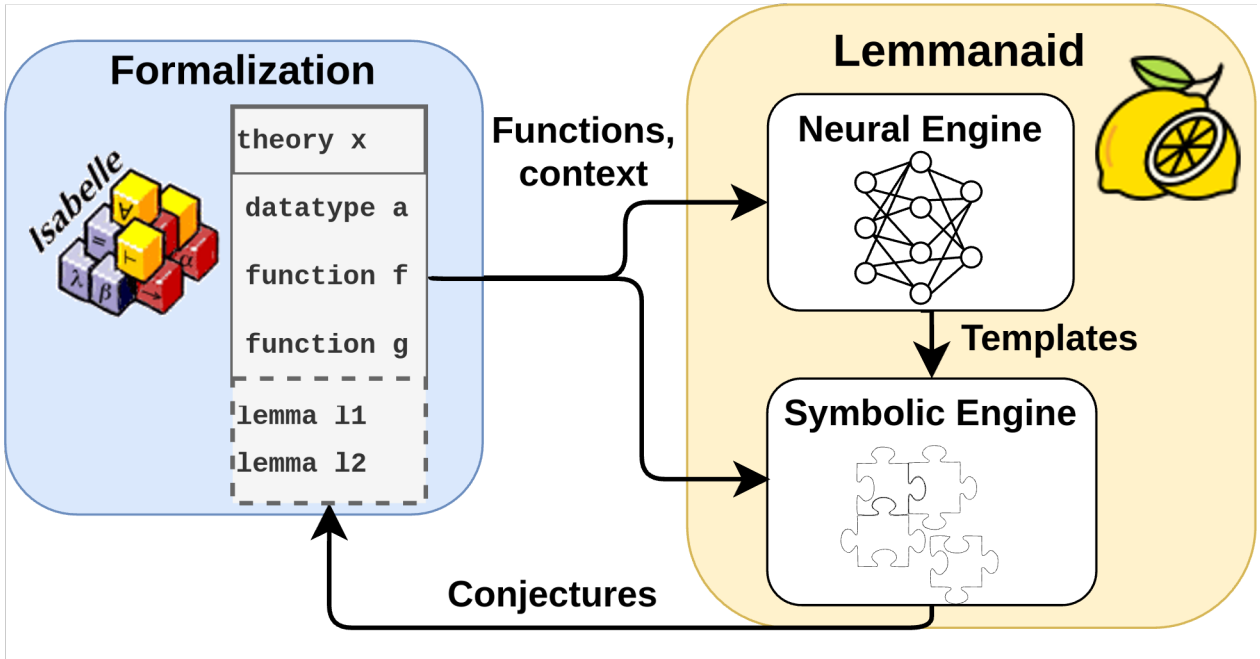


Figure 1: Left: Number of lemmas per template, sorted by frequency. Right: Cumulative percentage of lemmas in the dataset covered by most frequent templates.

	Template	# lemmas
1	?F (?G X Y) = ?H (?F X) (?F Y)	611
2	?F X = ?G (?H X)	566
3	X = ?F (?G X)	340
4	?F X = ?F (?G X)	280
5	X = ?F ?G X	247
6	?F (?G X Y) Z = ?H (?F X Z) (?F Y Z)	233
7	X = ?F X ?G	210
8	?F X (?G Y Z) = ?H (?F X Y) (?F X Z)	194
9	?F = ?G (?H X)	192
10	?F = ?G ?H X	184

Lemmanaid: Neuro-Symbolic Conjecturing



Lemmanaid: Neuro-Symbolic Lemma Conjecturing. Yousef Alhessi, Sólrún Halla Einarsdóttir, George Granberry, Emily First, Moa Johansson, Sorin Lerner, Nicholas Smallbone. Under review 2025. https://arxiv.org/abs/2504.04942

Results

How does Lemmanaid compare to:

- A neural model generating lemmas directly?
- A symbolic system (QuickSpec)
- What information should be included in the prompt?
- Beam search (size 4).
- Comparison of lemmas generated matching those in an (unseen) Isabelle formalisation.

	Lemma Success rate		
		HOL-train	
Method	HOL-test	AFP-Test	Octonions
Deepseek-coder-1.3b			
LEMMANAID (types + defs)	37.1%	21.6%	50.0%
LEMMANAID (types)	33.4%	22.5%	56.6%
LEMMANAID (defs)	31.3%	10.4%	38.6%
Neural (types + defs)	25.7%	10.4%	23.7%
Neural (types)	23.6%	13.8%	40.0%
Neural (defs)	21.5%	5.3%	21.1%
Combined	49.3%	30.9%	70.9%
QuickSpec	—	—	22.8%

Conclusion

- Neural and symbolic conjecturing have complementary features.
- Lemmanaid shows how to combine best of both:
  - LLM suggests where to do more detailed search.
  - Wider range of properties can be found.
- Even getting some lemmas will speed up formalisations in proof assistants.
- Next steps:
  - Experiments with other LLMs/proof assistants (computational resources…)
  - Apply counter-example checkers and automated proof tools to conjectures.
  - User studies.
  - Full integration with a proof assistant. Tool in proof engineer’s workflow.

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY