AI-Generated Proofs: The Deep Technical Reality Beyond the Headlines

The revolution in AI-generated mathematical proofs has accelerated dramatically since 2020, reaching an inflection point with AlphaProof's 2024 IMO silver medal. ResearchGate↗ But beneath these headline achievements lies a complex technical landscape of remarkable successes, fundamental limitations, and diverging approaches across proof assistant ecosystems. This report provides the technical depth your previous analysis lacks.

The comprehensive achievement landscape reveals unexpected patterns

While AlphaProof dominated headlines, the field's progress spans dozens of systems across five years, each revealing different technical bottlenecks and breakthroughs. The data tells a story of explosive recent progress shadowed by persistent fundamental challenges.

The 2020-2022 foundation period established viability through systems like GPT-f (56% on Metamath, September 2020) arxiv↗Medium↗ and CoqGym's infrastructure. GPT-f represented the first deep learning system to contribute proofs actually adopted by the formal mathematics community—finding shorter proofs for 2 theorems in set.mm. arxiv +2↗ Yet this "first contribution" milestone itself indicates how limited early success was. Proverbot9001 achieved 19.8% on CoqGym, Tactician +3↗ while PACT introduced expert iteration and proof artifact co-training.

The 2023 retrieval revolution changed the game fundamentally. LeanDojo/ReProver pioneered retrieval-augmented theorem proving, treating premise selection as a learned embedding problem rather than heuristic matching. GitHub↗ NeurIPS↗ Magnushammer demonstrated transformers could beat traditional symbolic methods decisively—59.5% versus Sledgehammer's 38.3% on PISA. When combined with neural provers, this jumped to 71%. arXiv +2↗ Draft-Sketch-Prove achieved 39.3% on competition problems by translating informal proofs to formal sketches, nearly doubling the 20.9% baseline. GitHub↗ arXiv↗

AlphaGeometry's domain-specific breakthrough (January 2024) solved 25 of 30 historical IMO geometry problems versus Wu's method at 10, approaching human gold medalist average of 25.9. The system trained on 100 million synthetic theorems—no human proofs required. Nature +2↗ AlphaGeometry 2 (January 2025) pushed this to 84% on all 2000-2024 IMO geometry problems with 88% problem type coverage. arXiv↗arXiv↗ The symbolic deduction engine combined with neural guidance created a neuro-symbolic architecture Substack↗ that would define the field's future. Nature↗

The synthetic data explosion of 2024 eliminated dependence on scarce human proofs. DeepSeek-Prover generated 8 million synthetic proofs, achieving 46.3% on miniF2F-test versus GPT-4's 23%. GitHub↗ On FIMO (148 IMO-level Lean problems), it solved 5 problems where GPT-4 solved zero. arXiv +2↗ LeanNavigator (2025) took this further: 4.7 million theorems from Mathlib4 state graph exploration—over 10× previous datasets at 1 billion tokens. arXiv↗

AlphaProof's July 2024 validation proved neuro-symbolic approaches work at competition level: 4 of 6 IMO 2024 problems solved including Problem 6 (only 5 human contestants solved it). The system used Gemini for natural language to Lean translation, then reinforcement learning on millions of self-play problems. Training continued during the competition on problem variations. Google DeepMind↗

2024-2025 state-of-the-art systems now achieve near-90% performance. DeepSeek-Prover-V2 (April 2025) uses 671B parameter DeepSeek-V3 for informal reasoning plus 7B model for formalization, reaching 88.9% on miniF2F-test at Pass@8192 and solving 49 of 658 PutnamBench problems. Synced↗arXiv↗ Graph2Tac (January 2024) achieved 26.1% on unseen Coq projects with online definition learning—86.7% on the Poltac package. Tactician +3↗ PALM hit 40.4% on CoqGym using GPT-3.5 with retrieval and repair. arXiv↗arXiv↗ Cobblestone achieved 48% on CoqGym100 with divide-and-conquer whole-proof generation. arXiv↗

Novel techniques discovered by AI include proof compression strategies (GPT-f shortened 23 Metamath proofs), arxiv↗ recursive decomposition patterns (DeepSeek-Prover-V2's subgoal factorization), and premise retrieval strategies that

outperform traditional relevance filters. The systems learned to prioritize certain tactic sequences—for instance, aggressive simplification before case analysis, or hypothesis unpacking as first steps. Graph2Tac's online learning mechanism adapts to project-specific proof styles in real-time, learning 1.5× better with definition task versus offline training. OpenReview ↗ Proceedings of Machine Learning Research ↗

Yet **the performance ceiling reveals the challenge**: most systems still fail on 50-70% of benchmark problems. The gap from miniF2F (Olympiad-level, 488 problems) to research mathematics remains vast. No AI has contributed meaningfully to solving major open conjectures. The Riemann Hypothesis, P vs NP, Birch-Swinnerton-Dyer—all show zero AI progress.

What has not worked reveals fundamental barriers that synthetic data cannot solve

The failures matter more than successes for understanding the technology's limits. Despite massive investment, several approaches have hit hard walls.

Autoformalization remains catastrophically difficult. Current state-of-the-art achieves just 25% accuracy on high-school problems and 13% on undergraduate-level mathematics (Azerbayev et al., 2023). This isn't a training data problem—it's fundamental to mathematical language structure. GPT-4 produces roughly 38 perfect formalizations out of 150 attempts. The 76% accuracy in reverse (informalization) versus 25% forward reveals inherent asymmetry: creating formal specifications from informal mathematics is qualitatively harder than paraphrasing formal statements.

Five systematic error types plague autoformalization. **Hallucinated references** constitute 30-40% of errors—LLMs cite theorems that don't exist, like "Theorem 3.2.1" in libraries lacking it. Pattern matching without semantic understanding creates plausible-sounding but invalid references. **Undefined terms** (20-25% of errors) occur when mathematical objects lack proper formal definition. The word "canonical" exemplifies this: in class field theory it appears without specifying which canonical form. Overloaded terms like "cohomology" have 10+ distinct formal meanings—singular, analytic de Rham, algebraic de Rham, étale, crystalline, Galois, rigid, group cohomology. **Syntactic errors** (15-20%) produce compilation failures despite semantic correctness, and BLEU scores fail to capture this—syntactic similarity doesn't imply logical equivalence. **Improper statements** (15-20%) are mathematically invalid like "2/3 $\in \mathbb{Z}$". **Non-logical inference** (10-15%) involves statements not following logically from premises, where gap-filling fails to generate intermediate steps.

The definition alignment problem has no clear solution. Same informal concept → multiple valid formal representations that are logically equivalent but syntactically different. Conversely, syntactically similar formal statements can have opposite meanings. "Connected component" has multiple formalizations with subtle differences. For natural numbers, `(n - m) + n = m` is "morally true" but technically false unless $n \geq m$. Context dependency pervades mathematical notation—operators get overloaded, subscripts and conditioning get dropped, subexpressions get elided with "...", context alone determines meaning. The expectation symbol in statistics requires determining which distribution from context with no explicit notation.

Data scarcity operates at catastrophic scale. Lean's mathlib contains 140,000 proofs. Isabelle's Archive of Formal Proofs has 250,000. Combined Proof Pile dataset: only 500MB total. arXiv ↗ Compare this to Python code on GitHub (billions of lines) or natural language (trillions of tokens). Formal proofs are orders of magnitude smaller. Current data is "decent for small models (billions of parameters)" but "insufficient for models with hundreds of billions of parameters." arXiv ↗ Standard scaling laws break down. Synthetic data generation creates infinite quantity but "complexity and quality often do not match human-written ones."

AI cannot discriminate interesting theorems from trivial ones—a fundamental limitation. An ATP can list millions of provable theorems from axioms, but nearly all are trivial and uninteresting. No current method filters for mathematical significance. This "would require a radical change in how the software functions" according to Pantsar (2024). Markuspantsar ↗ Humans formulate interesting conjectures through pattern recognition and intuition; AI systems cannot identify which problems are worth solving. Conjecturing remains "indispensable for mathematicians but currently limited in AI."

Where AI falls short compared to humans encompasses creative reasoning gaps. For proof planning, humans use high-level strategies while AI performs local search without global strategy. Cross-domain connections drive breakthroughs—

Wiles proved Fermat's Last Theorem by connecting elliptic curves and modular forms—but AI systems don't identify such connections. The Robbins Conjecture (solved by McCune's ATP in 1997) required "a proficient user"—not autonomous solution. Success remains "sporadic." arXiv↗ The Four Color Theorem and Kepler Conjecture involved computer-assisted proofs requiring years of human setup, Wikipedia↗ qualitatively different from autonomous proving. Current contribution is "negligible compared to human effort": seL4 OS kernel verification took 20+ person-years, CompCert C compiler verification took multiple PhD years.

Scalability problems stem from infinite action space. In Go and chess, finite action spaces allow exhaustive search where exploring "wrong" moves still provides learning signal. In theorem proving, infinite action space means "a dead end is just a dead end"—no partial credit. Computational effort is "simply wasted." Proof length can exceed the number of atoms in the known universe for small statements. Deep neural networks cannot currently predict elementary sums accurately: "What is 437+156?" remains an unsolved challenge for pure neural approaches. Minor changes in goal statement can impact provability dramatically.

Generalization failures appear consistently. The INT benchmark tests six generalization types, finding transformer models show "much larger out-of-distribution generalization gaps than GNNs" despite better in-distribution performance. Novel theorems requiring premises unseen during training—the realistic use case for mathematicians—show severe degradation. AlphaGeometry succeeds only in Euclidean plane geometry; "adapting entire system would require substantial redesign" for other domains. Nature↗ Performance degrades rapidly with difficulty: below AIME level shows some success, IMO level shows rare success (AlphaProof being recent exception), research mathematics shows essentially zero autonomous success.

Theoretical limitations provide hard boundaries. Gödel's incompleteness theorems guarantee any consistent theory with arithmetic has true unprovable statements. ATPs "will fail to terminate when statement is undecidable" even if true in the model—no algorithmic solution possible. Propositional logic is decidable but co-NP-complete with only exponential-time algorithms. First-order logic has semi-decidability: valid formulas are computably enumerable but invalid formulas cannot always be recognized. Many interesting mathematical theories are undecidable. Some proofs are "unbelievably large," exceeding universe-sized bounds for small statements.

Transformer limitations for reasoning remain an open question. Evidence suggests transformers "learn to reason only through grokking" (training far beyond overfitting), "fail to learn true reasoning generalizable across data distributions," and "fall short on planning tasks." arXiv↗ The memorization versus reasoning trade-off means transformers excel at mathematical facts but struggle with multi-step reasoning, long context handling, learning abstractions, and hierarchical planning.

Premise selection failures persist despite being well-studied. ReProver achieves 27.6% recall@10 on unseen premises versus BM25 baseline at 15.5%—still missing 70%+ of relevant premises. arXiv↗ NeurIPS↗ This "enduring challenge" has no clear solution path.

Expert iteration plateaus after few iterations. Gains diminish, not sustaining self-improvement. "Still an open problem to obtain continuous improvements." GPT-4 "made hopeless algebraic and numerical errors" and "very often what holds it back is high school algebra." It cannot reliably perform nested radical simplifications. arXiv↗ LLMs "struggle with intrinsic self-verification"—cannot reliably verify their own generations. Self-correction attempts often introduce new errors.

Coq's Calculus of Inductive Constructions creates unique AI challenges compared to Lean

Coq's dependent type theory foundation differs fundamentally from Lean, creating distinct opportunities and obstacles for AI integration.

CoqGym and ASTactic (2019) established the benchmark: 71K human-written proofs from 123 projects, 68,501 theorems total. arXiv↗ DeepAI↗ ASTactic used TreeLSTM to encode proof states as abstract syntax trees, University of Massachusetts↗ achieving 12.2% versus built-in Coq tactics at 4.9%. Combined with CoqHammer: 30%. The Morning Paper↗ The system generated tactics by composing ASTs in predefined grammar but was limited to Coq 8.9.

Proverbot9001 (2020) from UC San Diego used RNNs with manually engineered features, achieving 19.8% on CoqGym alone, 28% with CoqHammer. arXiv↗arXiv↗ On CompCert (major verified C compiler), it proved 27.5% of test theorems. This 4× improvement over ASTactic came from depth-limited search (typically depth 6, width 3) and better argument prediction. But version compatibility issues (only 8.10-8.12) and installation complexity prevented wide adoption.

CoqHammer (2018), though not neural, provides critical baseline. It translates Coq's CIC to untyped first-order logic for SMT solvers (Z3, CVC4, Vampire, E), achieving 26.6% on CoqGym and 40.8% on standard library bootstrapping. ResearchGate +4↗ The 20-second prover plus 5-second reconstruction timeout works well for goals "close to" first-order logic but cannot perform induction—a key limitation. The Morning Paper↗ It struggles with sophisticated dependent types and boolean reflection. CoqHammer↗

Tactician and Graph2Tac represent state-of-the-art. Tactician uses k-nearest neighbor with locality sensitive hashing for online learning from proofs in real-time. Springer +3↗ Graph2Tac (January 2024) introduced novel GNN architecture with hierarchical representation learning, achieving 26.1% with definition task versus 17.4% without. On Poltac package (unseen during training): 86.7% success. Tactician↗Tactician↗ The online definition task learns embeddings for new concepts never seen during training—critical for Coq's diverse ecosystem. Graph2Tac outperforms all other general-purpose Coq provers by at least 1.48×, better than CoqHammer, Proverbot9001, and transformer baselines. OpenReview↗Proceedings of Machine Learning Research↗

Recent LLM systems show dramatic improvements. PALM (2024) uses GPT-3.5 with retrieval augmentation and backtracking repair, achieving 40.4% on CoqGym—outperforming Passport (22.9%), Proverbot9001, and DSP. arXiv↗ arXiv↗ Cobblestone (2024) combines divide-and-conquer with GPT-4/Claude whole-proof generation and fail-safe execution to localize errors, achieving 48% on CoqGym100, 38% on coq-wigderson, using $1.25 per run at 14.7 minutes average. arXiv↗arXiv↗ With oracle information, it reaches 58%. arXiv↗

Coq's CIC creates unique AI challenges. The Calculus of Inductive Constructions' dependent types create proof states with complex type dependencies. Yale↗MIT Press Bookstore↗ Eliminators for inductive types can be intricate. Universe polymorphism adds complexity. Strict positivity checking can fail even with seemingly valid proof scripts. This makes encoding proof states harder than Lean's type theory, complicates premise selection due to type-level computation, and creates difficulty with proof terms involving sophisticated dependent pattern matching.

Ltac complexity makes Coq harder than Lean for AI. Coq's tactic language is more complex and less structured than Lean's system. Custom tactics in projects create project-specific proof styles. Tactics can have side effects and complex argument structures. Many accept compound expressions as arguments, not just simple identifiers. ASTactic was limited to predefined grammar subset. Custom tactics unseen during training cause failures.

The computational versus proof-relevant distinction unique to CIC creates errors. The distinction between Prop (proof-irrelevant) and Set/Type (computational) means you cannot eliminate from Prop to Set without restrictions. Cannot use pattern match on `or A B` to produce boolean. Yale↗ Models must understand this distinction; incorrect eliminations produce cryptic error messages. It's hard to learn from errors due to type system restrictions.

Coq versus Lean comparison reveals why Lean dominates recent research. Coq's fragmented ecosystem (standard library, math-comp, various projects with different proof styles) contrasts with Lean's centralized mathlib (~500K lines with consistent style). Wikipedia↗ Coq has 124 projects in CoqGym with varying approaches; Lean's monolithic mathlib provides consistent training data. Models trained on Coq standard library fail on math-comp-heavy projects. Lean's quotient types avoid "setoid hell" common in Coq. Lean's tactic system is more structured and compositional. Plugin API changes between Coq versions break tools; Lean 4 offers more stable API.

What works best for Coq: hybrid neural plus symbolic (hammer integration essential), online learning to handle project diversity, ensemble methods (Diva's success shows diversity helps), graph-based models for CIC's dependency structure, and LLM divide-and-conquer for complex proofs. Cobblestone shows 96-137% improvement with hammer integration. arXiv↗

Proof assistant ecosystems beyond Lean and Coq show divergent AI maturity

Isabelle/HOL leads in automation quality with Sledgehammer translating goals to untyped first-order logic for ATPs like Z3 and Vampire. arXiv↗ Stack Overflow↗ PISA (Portal to ISAbelle) contains 183K theorems and 2.16M proof steps from the Archive of Formal Proofs. Thor (2022) integrated language models with Sledgehammer, achieving 37.3% on MiniF2F-valid. arXiv↗ GitHub↗ Magnushammer's retriever hits 71% on PISA, 36.9% on MiniF2F-valid. GitHub↗ arXiv↗ IsaMini (2024) with redesigned proof language achieves 69.1% pass@1 on PISA, 79.2% pass@8.

Isabelle's higher-order logic foundation provides rich expressiveness, and the Archive of Formal Proofs' 250K proofs offer substantial training data. PISA enables programmatic interaction. Strong hammer-style automation with Sledgehammer solves most problems on competition-level mathematics. Stack Overflow↗ arXiv↗ The structured Isar proof language is more readable and declarative than tactic-based approaches. Research activity is very high with multiple 2024-2025 papers on neural theorem proving and autoformalization.

HOL Light established early benchmarks but shows less recent activity. HolStep dataset (2017) provided millions of proof steps for higher-order logic, benchmarking various ML models. HOList (2019) created environment for deep RL theorem proving. DeepHOL (2019) used deep RL for tactic-based proving trained on Flyspeck (Kepler conjecture formalization). The Flyspeck AI/ATP system (2014) achieved 39% automation on 14,185 Flyspeck theorems in push-button mode with 30-second timeout on 14-CPU workstation. arXiv +2↗

HOL Light's simple minimalist design based on OCaml uses higher-order logic with small trusted kernel. The Flyspeck project represents 12 years of human effort to formalize the Kepler conjecture. Limitations include smaller library than Isabelle (limiting training data), less active development of new AI tools, and the difficulty interfacing with HOL Light (HOList requires custom Python implementation). Activity level is moderate with foundational work done 2016-2020 but less recent progress than Isabelle or Lean.

Mizar represents 50+ years of formal mathematics with one of the largest libraries. The Mizar Mathematical Library uses natural deduction style closer to mathematical writing, based on Tarski-Grothendieck set theory. Mizar40 (2015) achieved 40% automation on MML version 1147 using 14-strategy portfolio in 30 seconds. ResearchGate↗ Mizar60 (2023) reached 75% automation in hammer setting on 50th anniversary—six times more proof data than Mizar40 with 638,293 unique proof dependencies. Single strongest ENIGMA method alone matched the full 2015 portfolio at 51.4%. A 95-strategy portfolio achieves 58.4% on development set. arxiv↗

ENIGMA for Mizar uses gradient boosted decision trees and graph neural networks with 3-phase architecture: parental guidance, multi-phase selection, GPU server mode. Symbol anonymization enables terminology-independent learning. Learning-based clause selection for E prover trained on 3+ million ATP proofs. Premise selection methods include k-NN with TF-IDF, naive Bayes with extended features, LightGBM in binary classification mode, GNN-based dependent selection, and ensemble methods with harmonic mean combination. arxiv↗

Mizar's first-order logic with set theory and natural deduction style, combined with very large library (60K+ theorems by MML 1147), makes it rich in mathematical content. MPTP export system enables ATP integration. Limitations include translation to first-order logic losing some information, steep learning curve, and smaller active community than Isabelle/Lean. Activity level shows sustained high engagement as major AI/ATP research platform for 20 years (2003-2023) with continuous improvements.

Metamath's minimalist design resembles "assembly language of proof assistants" with single substitution rule. Set.mm database proves 74 of "Formalizing 100 Theorems" with at least 19 proof verifiers. Wikipedia↗ GPT-f (2020) achieved 56.22% success rate on held-out test set versus MetaGen-IL baseline at 21.16%—the first deep learning system to contribute proofs accepted by formal mathematics community. It found new short proofs incorporated into main Metamath library using 128-step proof search with language modeling approach. arxiv +2↗

The Metamath Hammer (2023) translates to higher-order and first-order TPTP formats, with state-of-the-art ATPs proving 68% of Metamath problems with human-written premises—first hammer system for Metamath combining proof reconstruction and premise selection. Dagstuhl↗ Limitations include low-level nature making proofs verbose, 128-step search limit insufficient for complex proofs, less automation than systems like Isabelle, and smaller active community. Activity level is moderate with major work 2016-2020 with OpenAI and recent hammer development (2023).

Agda shows minimal AI integration. This dependently typed programming language based on Martin-Löf type theory has heavy reliance on metavariables for incremental construction and no separate tactics language—proofs written in functional style. AI work is very limited, primarily exploring external ATP integration rather than ML/neural approaches. Waldmeister integration (2011) connected equational theorem prover through Agda's reflection mechanism, but this isn't ML-based. No major neural theorem proving benchmarks have been published.

Agda's limitations for AI include no large corpus of formalized proofs for training, smaller community and library, no established ML/neural theorem proving infrastructure, metavariable-based approach not fitting tactic paradigm, and minimal research attention from AI/ML community. Activity level for AI is very low—occasional discussions but no major neural theorem proving efforts. Total functional programming language ensuring all programs terminate, with constructive mathematics foundation, but the smaller standard library and Unicode-heavy syntax limit accessibility.

Automation rate comparison shows Mizar leads at 75% (most mature AI/ATP integration over 20 years), Metamath at 68% with hammer and 56% pure neural (GPT-f), Isabelle at 50-70% depending on corpus, HOL Light at 39% on Flyspeck, HOL4 at ~50% with TacticToe, and Agda at minimal/none. For AI research suitability, Isabelle ranks highest with large corpus (250K proofs), strong tooling, and active community. Lean follows with rapidly growing mathlib. Mizar provides largest library with 20 years of AI/ATP research. Coq offers large corpus but complexity creates challenges.

Technical factors affecting AI integration show positive factors include large proof corpus (Isabelle, Mizar), higher-order logic allowing rich expressiveness, good ATP translations (Mizar first-order, Isabelle Sledgehammer), strong hammer tools, and structured proof languages (Isabelle Isar, Mizar declarative). Negative factors include small libraries limiting training data (Agda, MetaMath), complex type systems complicating neural encoding (Agda dependent types), low-level proofs too verbose (MetaMath), limited tooling for ML integration (Agda), and higher-order features difficult for ATPs (HOL Light).

Neural architectures and training methods reveal why some approaches dominate

AI proof systems comprise three essential components: neural model components (policy network predicting next tactics, value network evaluating proof state promise, premise retriever selecting relevant lemmas), integration layer (proof assistant interface, state manager, verification kernel), and search engine (tree search algorithms like MCTS or best-first, beam search, backtracking mechanisms).

Transformer-based architectures dominate modern systems. Encoder-decoder transformers like ByT5 and T5 process proof states plus retrieved premises through encoders with relative position encoding superior to absolute, typically 6-12 layers with 768-1024 hidden dimensions. Decoders generate tactics autoregressively with causal attention mask and cross-attention to encoder, outputting tactic tokens from vocabulary of 50K-100K. Key technical features include residual streams (sum of all previous layer outputs plus original embedding enabling interpretability through linear operations) and multi-head attention with 8-16 heads capturing different relationships. Transformer Circuits↗

GPT-f adapted GPT-2/3 architecture for Metamath, generating proof steps as text completion with 32 candidates per step, shortening 23 existing proofs. arxiv↗ DeepLearning.AI↗ DeepSeek-Prover-V2 (2025) uses 671B parameter model with subgoal decomposition, combining DeepSeek-V3 for informal reasoning with 7B model for formal proof, achieving 88.9% on miniF2F-test at Pass@8192. arXiv↗ Reasoning tokens (DeepSeek-R1, Kimina-Prover) generate internal "thinking" before formal tactics, aligning informal mathematical reasoning with formal steps.

Graph Neural Networks encode formulas as graphs with nodes representing operators/variables/constants with type embeddings and arity information, edges capturing parent-child relationships and argument positions. Variants include GraphSAGE (neighborhood aggregation with sampling), Graph Attention Networks (learned attention weights), and Message Passing Neural Networks. Three to five GNN layers typically aggregate via mean/max pooling with layer normalization. GNNs preserve structure, achieve name invariance (learning relationships independent of variable names), and handle large formulas efficiently. arXiv↗ arXiv↗ HOList (2019) pioneered GNNs for higher-order logic. NIAGRA developed name-invariant graph representations. NeuroTactic uses graph contrastive pre-training.

Supervised learning extracts (proof_state, tactic) pairs from existing proofs, training with cross-entropy loss between predicted and ground-truth tactics. Challenges include data scarcity (limited formal proofs available), imitation learning

ceiling (can't exceed teacher performance), and distribution shift (training distribution differs from deployment). Solutions involve data augmentation (proof step permutation, equivalent formulations) and synthetic data generation (DeepSeek-Prover generated 2B synthetic examples; MarkTechPost↗ LeanNavigator produced 4.7M theorems via state graph exploration).

Reinforcement learning formulates theorem proving as Markov Decision Process: state is proof state (goals, hypotheses, context), action is tactic application, reward is binary (1 if proof complete, 0 otherwise) or sparse at terminal states. Proximal Policy Optimization uses clipped surrogate objective preventing large policy updates in actor-critic architecture. Group Relative Policy Optimization optimizes based on relative ranking of multiple outputs, more stable than PPO for structured generation—used in DeepSeek-Math and ProofSeek, sampling K candidates, ranking by reward, updating based on relative performance.

Monte Carlo Tree Search balances exploration/exploitation via UCB1 formula: Score = Q(node)/N(node) + C × sqrt(ln(N(parent))/N(node)). arXiv↗ Theorem-proving optimizations include value network guidance replacing random rollouts, proof state evaluation predicting success probability trained on successful (label 1) versus failed (label 0) paths, and bigstep nodes storing only key decision points to reduce memory. DeepHOL, HOList, rlCoP, monteCoP, and DeepSeek-Prover-V1.5 all use MCTS variants.

Expert iteration generates proofs with current policy, filters successful proofs via verifier, adds them to training dataset, and fine-tunes policy on augmented data. This combines supervised learning stability with reinforcement learning exploration but plateaus at ~13% on LeanWorkbook. arXiv↗ OpenReview↗ Self-play plus expert iteration (STP, 2025) generates new conjectures via conjecturer model producing challenging but provable theorems, with prover attempting proofs and updating both models. This doubles performance: 26.3% versus 13% on LeanWorkbook. arXiv +2↗

DeepSeek-Prover-V1.5 combines RL with MCTS and proof assistant feedback directly in training loop—verifier-in-the-loop only rewarding verified proofs. GitHub↗ DEV Community↗ Leanabell-Prover-V2 integrates verifier with long chain-of-thought, multi-turn verifier interactions, and feedback token masking for stable training with self-aware error correction. arXiv↗ arXiv↗

Reinforcement learning versus supervised learning trade-offs show supervised requires high data (needs expert proofs) but offers stable reliable training with high sample efficiency, limited by training data ceiling, no exploration, best for fast prototyping with limited compute. RL requires lower data (learns from interaction), can exceed human performance, but suffers unstable training requiring careful tuning, low sample efficiency with many failed attempts, active exploration discovering new strategies, best for long-term performance and novel proofs. Expert iteration bridges the gap with +40% relative improvement. Self-play plus expert iteration breaks plateaus, doubling expert iteration performance (26% vs 13%) though requiring more complex implementation.

Best-first search maintains priority queue ordered by heuristic score, popping highest-priority states, generating tactics with policy network, scoring new states with value network, and repeating until proof found or timeout. arXiv↗ This efficiently explores most promising paths first with parallelizable branches and interpretable priority ordering. Heuristics include value network predictions, goal count reduction, and syntactic simplicity metrics. GPT-f baseline, LeanDojo's ReProver, and many tactic-step systems use this approach.

Hypertree proof search enables distributed asynchronous proof search with multiple agents exploring concurrently, policy and critic networks updated collectively, and backpropagation through hypertree structures. This allows asynchronous parallel exploration with shared neural network updates efficiently exploring large proof spaces.

Proof assistants differ fundamentally in AI suitability. Dependent types (Lean, Coq) encode rich specifications but create more complex inference. Higher-order logic (Isabelle/HOL) enables classical reasoning closer to mathematical practice but offers less precise specifications. Isabelle's Sledgehammer provides strongest automation calling external ATPs with 71% success using neural guidance. arXiv +2↗ Coq has moderate automation with Omega, congruence, ring tactics and Ltac for custom tactics (powerful but complex). Lean has less built-in automation requiring more manual proof construction but this provides clearer signal for neural models.

Lean 4's advantages explain why 75% of 2024-2025 papers use it: clean separation of proofs versus programs with definitional proof irrelevance and predictable equality, modern design with efficient kernel and better metaprogramming,

active community with 1000+ mathlib PRs yearly and consistent style, quotient types built-in with reduction rule avoiding setoid hell, and dependent pattern matching more flexible than Coq with generalized congruence closure.

Research frontiers show the field at an inflection point

The field experienced near-doubling of publications annually in 2024-2025, with AlphaProof's success catalyzing emphasis on formal mathematical reasoning over purely informal approaches. ResearchGate↗ Major trends include synthetic data generation, neurosymbolic integration, LLM-enhanced proof search, and connections to AI safety through formal verification.

Princeton's Goedel-Prover-V2 (January 2025) represents strongest open-source theorem prover featuring self-correction mode for iterative proof improvement. Led by Chi Jin, Yong Lin, and Sanjeev Arora's team, it shows strong results on PutnamBench, miniF2F, and MathOlympiad benchmarks with Lean4 integration for verified correctness.

DeepSeek-Prover-V2 (April 2025) introduces recursive theorem-proving pipeline with cold-start training from DeepSeek-V3, achieving 88.9% pass ratio on miniF2F-test and solving 49 of 658 PutnamBench problems. ProverBench provides new benchmark with 325 problems including recent AIME competitions.

LeanNavigator (2025) generates 4.7 million theorems (1 billion tokens) from Mathlib4 by exploring state transition graphs —over 10× previous datasets. LeanProgress (February 2025) predicts remaining proof steps with 75.1% accuracy, improving search efficiency by 3.8% on Mathlib4. TheoremLlama achieves 36.48%/33.61% on MiniF2F-Valid/Test, surpassing GPT-4's 22.95%/25.41%.

ICLR 2025 accepted papers include Lean-STaR (learning to interleave thinking and proving), ImProver (agent-based automated proof optimization), CARTS (diversified tactic calibration and bias-resistant tree search), FormalAlign (automated alignment evaluation for autoformalization), Herald (natural language annotated Lean 4 dataset), and ProofAug (fine-grained proof structure analysis achieving 52.5% on miniF2F-test).

Unsolved challenges actively tackled include data scarcity (formal proof data orders of magnitude smaller than programming/NL data with Proof Pile only 500MB versus trillions of tokens for LLM training), autoformalization at scale (no automatic metric correlates with human evaluation; syntactic similarity doesn't equal logical equivalence), and proof search efficiency (open questions comparing search versus whole-proof generation, small models with more exploration versus large models with less, MCTS versus best-first search, CPU versus GPU optimization contexts).

Long context and complex proofs challenge systems on proofs requiring over 1000 steps or spanning multiple files. MiniCTX benchmark (August 2024) tests context-dependent proving with tens of thousands of tokens. Hierarchical decomposition approaches (POETRY, DSP) and library learning with modular proof strategies address this.

GPT-4, Claude, and Gemini integration shows neurosymbolic AI consensus post-AlphaProof: pure neural or pure symbolic approaches are insufficient. AlphaProof uses Gemini for natural language to formal language translation with AlphaZero RL algorithm and self-play mechanism proving/disproving millions of statements. COPRA uses frontier LLMs with error messages as prompts. GPT-4 baseline achieves 23% on miniF2F, substantially improved by specialized systems. Retrieval-augmented generation with ReProver pioneered RAG for theorem proving. Lean-STaR interleaves informal thinking with formal proving.

Synthetic data generation represents major innovation with AlphaGeometry's 100M synthetic geometry theorems (no human pretraining), LeanNavigator's 4.7M theorems from state space exploration, DeepSeek-Prover's iterative quality improvement through proof/disproof attempts, and scale assurance via parallel proving of negated statements to avoid unprovable traps.

Whole-proof versus step-by-step generation shows emerging trend toward whole-proof with Baldur's direct generation without search, DeepSeek-Prover generating complete proofs truncating at first error, offering lower latency for interactive applications but less exploration versus faster results trade-off.

Library learning and abstractions include LEGO-Prover proposing reusable lemmas during proving, tactic induction learning domain-specific proof strategies (Peano, LEMMA), progressive abstraction building mirroring human learning, and

lemma mining from existing corpora.

Grand challenges include autonomous theorem proving (discriminating interesting theorems from trivial ones—cannot identify mathematical significance), research-level mathematics (proving Millennium Prize-level problems with current success mainly at high-school/undergraduate level), cross-domain generalization (generalist versus specialist trade-offs), hierarchical reasoning (decomposing high-level goals with current limitations in LLM decomposition), evaluation and benchmarking (no unified evaluation across proof assistants), human-AI collaboration (usability bottleneck for mathematician adoption requiring HCI methods), and autoformalization quality (no automatic metrics aligned with human judgment).

Five-year predictions suggest near-term (2025-2026) will bring 10-100× increase in synthetic formal data, autoformalization of major textbook corpus, cross-language translation becoming routine, routine IMO gold medal problem solving, undergraduate-level theorem proving at scale, and interactive proof assistants as mainstream developer tools. Mid-term (2027-2029) may achieve AI assistance in proving research-level conjectures, collaborative human-AI formalization projects, discovery of novel mathematical connections, routine verification of critical software/hardware, AI-generated code with correctness proofs, dramatic cost reduction from 20+ person-years to months, and formal verification of AI-generated scientific hypotheses.

Connections to AI safety through formal verification appear in Guaranteed Safe AI frameworks requiring world model, safety specification, and verifier—though formalizing "don't let humanity go extinct" faces significant technical obstacles. Proof-carrying code for AI envisions systems providing mathematical proofs of safety before actions with provably compliant hardware and provable contracts, though modeling physical/social systems as formal systems remains challenging. Current applications include autonomous vehicle control (Lyapunov function verification), neural network verification (ReLU activation handling), cryptographic implementations, and AWS S3 policy verification.

EuroProofNet coordinates European research on formal proofs through conferences (AITP, ITP, TYPES, CADE) focusing on proof system interoperability and LLM integration. Major 2025 venues include AITP 2025 (Aussois, August 31-September 5), ITP 2025 (Reykjavik, September 27-October 3), and Edinburgh Workshop on Theorem Proving and Machine Learning in the Age of LLMs (April 7-8).

The field transitions from "can AI do theorem proving?" to "how do we scale AI theorem proving to research mathematics and real-world verification?" Key bottlenecks are data scarcity, autoformalization quality, and human-AI interaction design. The next five years will likely see routine IMO gold performance, undergraduate-level formal mathematics at scale, dramatic reduction in formal verification costs, and first AI contributions to research mathematics. The grand challenge remains developing AI that can identify interesting mathematics, not just prove given theorems—requiring fundamental advances in mathematical creativity and significance assessment beyond current capabilities.