STATPROVER TECHNICAL REPORT

StatAI Lab
School of Statistics and Data Science
Shanghai University of Finance and Economics
Shanghai, China
zhoufan@mail.shufe.edu.cn

April 19, 2026

ABSTRACT

Although large language models (LLMs) show remarkable capabilities, they still struggle with rigorous mathematical derivations in statistics. Recent benchmarks, such as StatEval, systematically revealed these limitations in research-level tasks. To bridge this gap, we introduce StatProver: an LLM-based automated proof generation and self-correction system. StatProver integrates proof generation and multi-turn self-correction into a six-stage pipeline, supporting both automated and human-in-the-loop workflows. To ensure logical robustness, the system features a dynamic framework refinement mechanism that aligns initial proof logic with retrieved expert-verified proof frameworks. Furthermore, we propose a novel, data-driven snippet-level self-correction module. By analyzing LLM failure trajectories across 40,366 StatEval problems, we constructed a large-scale fault database. During inference, StatProver dynamically segments generated proofs, matches them against historical faults, and executes targeted corrections. Empirical evaluations demonstrate that StatProver produces more accurate proof frameworks, effectively rectifies reasoning errors, and ultimately yields high-quality mathematical proofs. StatProver is publicly available at https://statprover.com/.

1 Introduction

LLMs have established a strong foundation in understanding and generating text across diverse disciplines. Foundational architectures, such as GPT-5.4 [1], Claude Opus 4.6 [2], Llama 4 Maverick [3], and Gemini 3.1 Pro [4], show this by distilling vast amounts of human knowledge into their parameters. Building upon this broad linguistic competence, the recent research focus has shifted toward enhancing their multi-step problem-solving abilities. The development of reasoning-specialized models—optimized through reinforcement learning and scaled inference-time compute, notably the GPT-5.4 Thinking variants and Qwen 3.6-Plus [5]—illustrates this trajectory. Together, these advancements indicate that LLMs are gradually transitioning from general pattern matching to deliberate, sequential reasoning.

This transition toward sequential reasoning is most prominently showcased in the domain of formal mathematics and logical deduction. The continuous emergence of challenging modern benchmarks—shifting from earlier foundational datasets toward expert-level evaluations like FrontierMath [6], and the Olympiad-suite Omni-MATH [7]—has charted the rapid baseline progress of these models. However, the most striking breakthroughs extend far beyond high scores on static datasets. In true tests of apex mathematical intellect, recent AI systems have successfully solved actual problems from the International Mathematical Olympiad (IMO) under competition conditions, achieving definitive gold-medal-level performance [8, 9]. Parallel to these competitive milestones, novel frameworks leveraging meta-context engineering and agentic skill evolution [10] have enabled LLMs to dynamically refine their reasoning strategies within complex logical environments. This growing competence is also reshaping the workflow of theoreticians; distinguished mathematicians are increasingly adopting LLMs as interactive co-pilots to explore advanced concepts [11, 12].

Despite the progress in formal mathematics, the capability of LLMs in advanced statistical derivation remains limited. Statistical proofs differ from standard mathematical theorem proving in analytical tools and objectives. While pure mathematics often focuses on deterministic structures and exact logical equivalences, research-level statistics deals with stochastic processes. This involves the derivation of asymptotic convergence rates, the establishment of probabilistic

error bounds, and adherence to complex regularity conditions, such as the positive definiteness of high-dimensional matrices. Due to these substantial differences in domain knowledge and reasoning patterns, the proficiency LLMs exhibit in formal mathematics does not naturally transfer to advanced statistics. Recent benchmarks, such as StatEval [13], have evaluated state-of-the-art models in this area, showing their limitations in solving research-level proofs. Therefore, a dedicated framework is needed to improve LLM reasoning capabilities for statistical problem-solving.

To bridge this capability gap, we introduce StatProver, an automated statistical proof assistant specifically designed for rigorous research-level derivations. Unlike the direct application of LLMs, which treats proof generation as a one-shot, end-to-end "black-box" task, StatProver is structured as a modular, six-stage pipeline that integrates multi-level self-correction as a core procedural stage. By leveraging a multi-model orchestration strategy—utilizing GPT-5.4 for complex reasoning, Gemini 3.1 Lite for semantic intent parsing, and vector representations from text-embedding-3-large to facilitate similarity-based retrieval and filtering mechanisms—StatProver establishes a robust workflow for formal AI assistance. This design transforms the system from a passive generator into a collaborative research co-pilot, introducing a flexible Human-in-the-Loop (HIL) workflow that allows researchers to guide the derivation and verify formal problem assumptions at critical decision checkpoints.

The technical architecture of StatProver is driven by two core innovative mechanisms designed to tackle hallucinations from both macro and micro perspectives. First, the Dynamic Framework Refinement module anchors the proof process. Instead of generating text from a blank slate, StatProver extracts the mathematical essence of a problem using a novel bi-directional max-matching similarity algorithm to effectively overcome semantic dilution. It then aligns the proof's macro-skeleton with expert-verified paths retrieved from a curated database of over 40,366 statistical problems. Second, the Data-Driven Snippet-Level Self-Correction module acts as a logical guardian. By exhaustively tracking the failure trajectories of LLMs across all research-level tasks in the StatEval benchmark, we constructed a first-of-its-kind empirical fault repository. During generation, StatProver dynamically segments proofs into minimal logical units, diagnoses them against historical failures, and executes surgical, delayed fixes to ensure both local micro-level rigor and global macro-level coherence.

In summary, the key contributions of this work are as follows:

- **Retrieval-Driven Proof Framework Refinement:** We employ a bi-directional max-matching algorithm and an LLM-as-a-Judge mechanism to retrieve optimal reference frameworks. This dynamically refines the initial draft into a logically robust macro-skeleton, preventing early trajectory drift.

- **Data-Driven Snippet-Level Self-Correction:** Leveraging a large-scale repository of empirical LLM reasoning failures, we introduce a dynamic self-correction mechanism to identify and surgically rectify subtle micro-level logical leaps that standard self-reflection methods fail to detect.

- **Interactive Proof Assistant Platform:** We present StatProver, a modular six-stage pipeline that supports both end-to-end automated proof generation and flexible HIL workflows. This system is officially deployed and publicly accessible at `https://statprover.com`.

2 System Architecture and Workflow

In order to address the logical hallucinations and reasoning breaks encountered by LLMs in rigorous statistical derivations, StatProver employs a systematic six-stage automated pipeline. The system is designed with both modularity and flexibility, supporting not only end-to-end automated processing but also human-in-the-loop interventions at critical nodes. This section details the overall architecture of StatProver and its underlying core innovative mechanisms.

2.1 Overview of the StatProver Pipeline and Human-in-the-Loop Mechanism

StatProver operates as a systematic multi-model orchestrated pipeline designed to deconstruct rigorous statistical derivations into manageable and verifiable units. While different agents are assigned specialized roles—Gemini 3.1 Lite for keyword extraction and semantic intent parsing, text-embedding-3-large for generating high-dimensional vector representations of these keywords, and GPT-5.4 as the core reasoning engine—the system's architecture emphasizes modular flexibility. For clarity, we categorize the six operational stages of StatProver into three functional phases: Initialization, Retrieval-Driven Framework Refinement, and Generation and Self-Correction.

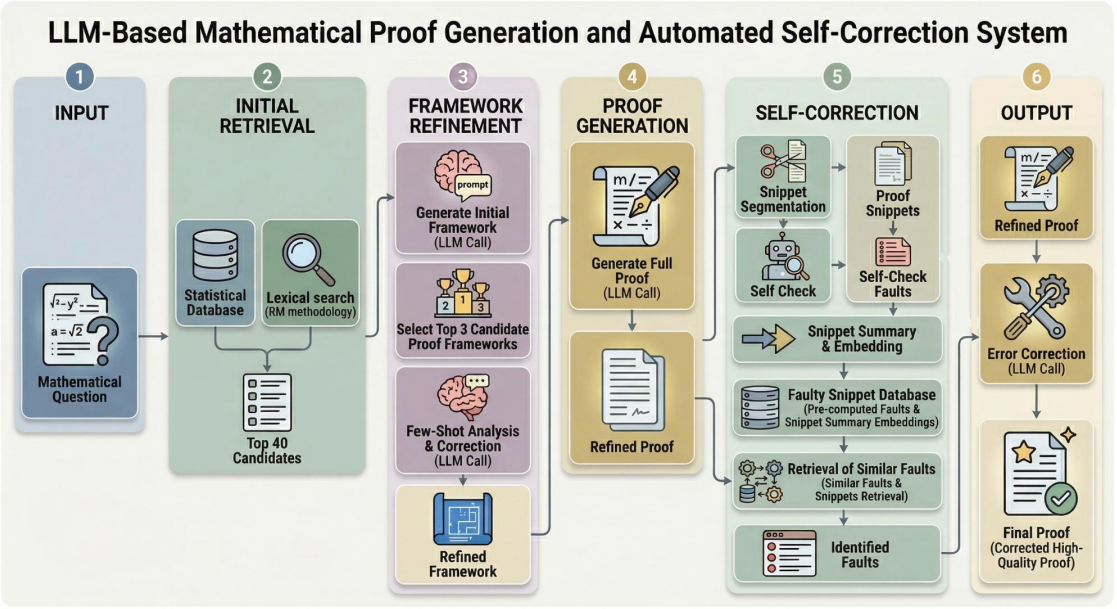

Figure 1: The six-stage automated pipeline architecture of StatProver.

Phase I: Problem Formalization

Step 1: Interactive Problem Refinement. The system first scans the input to find missing definitions or assumptions. Users can add these details manually or let the LLM auto-complete them. This process produces a structured "Complete Problem Description" that explicitly lists the notations, assumptions, prerequisites, and the proof goal.

Phase II: Retrieval-Driven Framework Refinement (Detailed in Section 2.2)

Step 2: Candidate Retrieval and User Verification. After extracting core keywords from the problem, the system utilizes text-embedding-3-large to generate their vector representations. By calculating cosine similarities based on these keyword embeddings, it identifies the Top-40 candidates from StatEval problem database. GPT-5.4 then narrows these to the 5 most relevant cases, which are presented to the user for verification as final reference candidates.

Step 3: Dynamic Framework Refinement. GPT-5.4 first evaluates these verified candidates to determine if their proof frameworks hold any actual reference value for the current problem. If instructive methodologies are identified, the system adaptively draws inspiration from these mature research strategies to review and refine its initial zero-shot proof draft, establishing a robust macro-skeleton.

Phase III: Generation and Self-Correction (Detailed in Section 2.3)

Step 4: Draft Proof Generation. Based on the refined framework and formalized problem description, GPT-5.4 executes detailed mathematical derivations to construct a complete, step-by-step draft of the proof.

Step 5: Data-Driven Snippet-Level Diagnosis. The system segments the proof into minimal logical units and matches them against our "Faulty Snippet Repository"—compiled by extracting standard error patterns from LLM failure trajectories across research-level problems in the StatEval benchmark—to accurately identify and record potential hallucinations or logical leaps.

Step 6: Global Correction and Final Output. By processing the original draft alongside the recorded errors, GPT-5.4 executes targeted surgical rectifications. The system then consolidates the corrected reasoning steps to output a mathematically rigorous and structurally complete final proof, formatted to professional academic standards.

2.2 Dynamic Framework Refinement

As outlined in Phase II of our pipeline in Section 2.1, a fundamental bottleneck in LLM-driven mathematical reasoning is the compounding effect of early logical missteps. In rigorous statistical derivations, if the initial proof skeleton is flawed, subsequent mathematical operations will inevitably fail. To eliminate these logical hallucinations at the source, StatProver introduces a retrieval-augmented Dynamic Framework Refinement mechanism. This mechanism guides the reasoning process through a two-stage algorithm: anti-dilution retrieval and truth-guided framework inspiration. This section details the technical execution of Steps 2 and 3.

Anti-Dilution Retrieval via Bi-directional Max-Matching To capture the precise mathematical essence of a problem, we avoid using global sentence embeddings. Verbose phrasing often causes semantic dilution in these global vectors.

Instead, we utilize Gemini 3.1 Lite to extract a dual-perspective feature set for each problem: 3–8 lexical keywords directly from the text and 3–8 semantic keywords summarized from the underlying logic.

To calculate the similarity between a target problem A and a candidate B from the comprehensive StatEval database, we employ an asymmetric Bi-directional Max-Matching algorithm. For each keyword embedding $e_{a,i}$ in A, generated via text-embedding-3-large, we find its maximum cosine similarity against all keyword embeddings $e_{b,j}$ in B. We average these maximum values to determine the directed similarity $\text{Sim}(A \to B)$:

$$\text{Sim}(A \to B) = \frac{1}{|A|} \sum_{i=1}^{|A|} \max_j \left(\cos(e_{a,i}, e_{b,j})\right) \tag{1}$$

The final comprehensive similarity $S(A, B)$ is the mean of the two bi-directional scores:

$$S(A, B) = \frac{\text{Sim}(A \to B) + \text{Sim}(B \to A)}{2} \tag{2}$$

By prioritizing the strongest local matches rather than an average pool, this method ensures that core statistical methods are not overshadowed by common terminology. This allows us to accurately isolate the Top-40 most relevant candidates.

LLM-as-a-Judge and Truth-Guided Framework Inspiration To convert the retrieved candidates into actionable guidance, we implement a contrastive reasoning mechanism powered by GPT-5.4. This process goes beyond simple template matching by focusing on strategic gap analysis:

- **Multi-Dimensional Assessment as the Judge:** GPT-5.4 evaluates the Top-40 candidates by comparing the target problem and its initial zero-shot draft against the candidate problems and their expert-verified frameworks. While similarities in mathematical background or proof goals offer initial hints, the model strictly assesses whether a candidate's actual proof structure provides actionable reference value. This rigorous filtering isolates up to three highly instructive reference cases.
- **Contrastive Gap Analysis for Inspiration:** To refine the initial proof draft, the system juxtaposes it against these verified reference frameworks. GPT-5.4 performs a strategic gap analysis to identify missing logical components. The model might adopt a highly suitable intermediate step from a reference or recognize the need for a case-by-case analysis that the initial draft overlooked. Crucially, the intrinsic information of the target problem remains strictly dominant. The system selectively integrates these methodological insights without being derailed by the reference paths. This ensures the construction of a logically robust macro-skeleton before granular derivation begins.

2.3 Data-Driven Snippet-Level Self-Correction

As outlined in Phase III of our pipeline (Section 2.1), while dynamic framework refinement establishes a logically sound starting point, statistical proofs often span multiple pages. As the reasoning chain lengthens, LLMs are prone to "attention decay," manifesting as micro-level hallucinations or logical leaps. To maintain accuracy throughout the generation cycle, StatProver implements a Snippet-Level Self-Correction module. This module operates through a closed-loop pipeline of feature retrieval, diagnostic characterization, and global targeted refinement powered by an empirical fault database. This section details the technical execution of Steps 4, 5, and 6.

Construction of the Empirical Fault Repository The corrective mechanism of StatProver is built upon large-scale empirical data. We tasked GPT-5.4 with solving the entire StatEval benchmark. By analyzing the model's failure trajectories across these statistical problems, we extracted over 80,000 fault snippets and structured them into a repository. Each entry forms a diagnostic triplet: the faulty proof text, an explicit error description, and the correct derivation extracted directly from the ground-truth solution. This provides the system with a case-based knowledge base of statistical pitfalls, reducing reliance on standard self-reflection prompts.

Semantic Dimensionality Reduction and Snippet Diagnosis During draft generation and subsequent diagnosis in Steps 4 and 5, a technical challenge is the noise introduced by dense LaTeX formulas, which often causes feature confusion during vector embedding. To resolve this, StatProver employs a semantic dimensionality reduction. The proof draft is dynamically segmented into minimal logical structure units. We define each unit as a conceptual reasoning triplet comprising the applied condition, the chosen mathematical method, and the resulting intermediate conclusion.

For each generated snippet, the system utilizes an LLM to summarize the core mathematical action into a plain-text semantic sentence. This summary is converted into a high-dimensional embedding via text-embedding-3-large to calculate cosine similarities against the fault repository. By matching semantics rather than raw symbols, the system

retrieves the Top-5 historical faults sharing the same underlying logic. GPT-5.4 then acts as a diagnostician, comparing the current snippet against these retrieved errors. If a logical flaw is confirmed, the system employs a deferred rectification strategy. It avoids immediate local fixes that might disrupt the global context, opting instead to record the error and the corresponding ground-truth paradigm into a diagnostic manifest.

Delayed Global Correction and Final Output Once the entire proof has been scanned in Step 6, the system executes a global targeted fix. The complete proof draft and the diagnostic manifest are fed back into GPT-5.4. Guided by the correct paradigms in the manifest, the model performs targeted replacements of the faulty nodes. This delayed strategy ensures that micro-corrections are executed with awareness of the overarching proof structure, preserving both local rigor and global coherence to produce a valid final proof.

3 Case Study: Deconstructing a Statistical Derivation

To demonstrate the practical efficacy of StatProver, we present a continuous, step-by-step case study tracking a single statistical derivation involving Directed Acyclic Graphs (DAGs) through our entire pipeline. This section is structured to mirror the workflow introduced in Section 2.1. Specifically, Section 3.1 introduces the chosen target problem. Section 3.2 illustrates how Dynamic Framework Refinement intervenes to correct the baseline LLM's macro-level structural hallucinations. Finally, Section 3.3 details how Snippet-Level Self-Correction identifies and rectifies micro-level logical leaps that emerge during the subsequent granular proof generation.

3.1 An Example Problem

The selected task requires proving two topological properties based on a specific partition of variables and a custom module construction rule. The complete problem description provided to the system is as follows:

> **Example Problem:**
> *Let $G = (\Psi', \mathcal{E})$ be a directed acyclic graph (DAG) where $\Psi' = (X, \Theta)$ represents the set of all variables, consisting of observable random variables X and parameters Θ. Suppose we have three modules Ψ'_A, Ψ'_B, and Ψ'_C formed according to Rule 1 (Constructing self-contained modules), and let $\Psi'_T = \Psi'_B \cup \Psi'_C$. Define the following partitioned sets: $\Psi_{BC} = (\Psi'_B \cap \Psi'_C) \setminus \Psi'_A$, $\Psi_{\underline{B}} = \Psi'_B \setminus (\Psi'_A \cup \Psi'_C)$, and $\Psi_{\underline{C}} = \Psi'_C \setminus (\Psi'_A \cup \Psi'_B)$. Let X^*_B and X^*_C be the observable variables of primary interest used to initialize modules B and C respectively. Let $\Psi'_S = (\Psi'_A \cup \Psi'_B \cup \Psi'_C)^c$ be the set of variables not contained in any module. Given that for any module M formed by Rule 1, $ch(\Psi'_{M^c}) = \emptyset$ and every variable in M has a directed path to an observable in X^*_M, prove that:*
> *(1) if $\Psi_{BC} \neq \emptyset$, there exists at least one edge $a \to b$ where $a \in \Psi_{BC}$ and $b \in (\Psi_{\underline{B}} \cup X^*_B) \cup (\Psi_{\underline{C}} \cup X^*_C)$*
> *(2) if $\Psi_{BC} = \emptyset$, then $(\Psi'_B \perp \Psi'_C \mid \Psi'_A)$.*

3.2 Impact of Dynamic Framework Refinement

When processed zero-shot, a baseline LLM struggles to construct a rigorous topological representation from flat textual tokens. Instead, it relies on heuristic text pattern matching, generating the following five-step initial proof framework.

> **Initial Zero-Shot Framework (Baseline):**
> 1. *Formulate the partition of $\Psi_B \cup \Psi_C$ into the overlap outside Ψ_A, the portion unique to B, and the portion unique to C, and relate these sets to the module-construction property that every node in a module has a directed path to a distinguished observable in that module.*
> 2. *Establish that any node $a \in \Psi_{\underline{BC}}$ must lie on some directed path to an observable in X^*_B and on some directed path to an observable in X^*_C, because a belongs to both modules Ψ_B and Ψ_C.*
> 3. *Derive, for the case $\Psi_{\underline{BC}} \neq \emptyset$, that along at least one such directed path starting from a node in $\Psi_{\underline{BC}}$, the first edge leaving the overlap must be of the form $a \to b$ with $a \in \Psi_{\underline{BC}}$ and $b \in (\Psi_{\underline{B}} \cup X^*_B) \cup (\Psi_{\underline{C}} \cup X^*_C)$, since the path must eventually reach a target observable outside the overlap.*
> 4. *Characterize the case $\Psi_{\underline{BC}} = \emptyset$ by showing that then $\Psi_B \cap \Psi_C \subseteq \Psi_A$, so every common node between the two modules is contained in the conditioning set Ψ_A, while nodes unique to one module cannot be descendants of nodes outside that module because $ch(\Psi_{M^c}) = \emptyset$ for Rule-1 modules.*

5. *Combine the absence of overlap outside Ψ_A, the DAG acyclicity, and the self-contained module property to conclude that any path connecting a node in Ψ_B to a node in Ψ_C must pass through Ψ_A, yielding $\Psi_B \perp \Psi_C \mid \Psi_A$.*

Our internal diagnosis reveals two structural failure modes in this baseline generation. The first is skipping logical steps. In step 3, the model assumes the path must eventually reach a target observable but skips the strict graph-theoretic justification for why a directed path must exit the overlap region and enter the specific subsets. The second is jumping to conclusions. In steps 4 and 5, the model infers conditional independence directly from the disjointness of the sets. It bypasses the mandatory d-separation path analysis and fails to rigorously invoke the closure property.

Following the system pipeline, StatProver queries the StatEval database to retrieve structurally analogous proofs. By juxtaposing the initial zero-shot draft against the reference frameworks extracted from these true proofs, GPT-5.4 conducts a contrastive gap analysis. The retrieved references demonstrate a consistent methodological constraint, noting that standard reference frameworks establish boundary conditions and structural invariants prior to initiating any path-based reasoning. Applying this acquired insight, the system restructures its initial logic to produce the following refined proof framework, which inherently resolves the structural deficiencies observed in the baseline generation.

Refined Proof Framework (StatProver):

1. *Set up the decomposition of $\Psi_B \cup \Psi_C$ into $\Psi_{\underline{BC}}$, $\Psi_{\underline{B}}$, $\Psi_{\underline{C}}$, and Ψ_A, and record the Rule-1 facts that every node in a module has a directed path to an observable in its initialization set and that no node outside a module can be a child of a node inside that module.*
2. *For any $a \in \Psi_{\underline{BC}}$, use membership in both Ψ_B and Ψ_C to obtain a directed path from a to some observable in X_B^* and a directed path from a to some observable in X_C^*.*
3. *Prove claim 1: along at least one such path from $a \in \Psi_{\underline{BC}}$, identify the first vertex reached outside $\Psi_{\underline{BC}}$; use the prohibition on edges from module nodes to nodes outside the module to exclude Ψ_S, and conclude that the corresponding first exit edge must be $a' \to b$ with $a' \in \Psi_{\underline{BC}}$ and $b \in (\Psi_{\underline{B}} \cup X_B^*) \cup (\Psi_{\underline{C}} \cup X_C^*)$.*
4. *Prove the structural consequence of $\Psi_{\underline{BC}} = \emptyset$: then $(\Psi_B \cap \Psi_C) \subseteq \Psi_A$, so outside Ψ_A the modules Ψ_B and Ψ_C are disjoint; moreover, by the Rule-1 closure property, a directed path starting in $\Psi_B \setminus \Psi_A$ cannot leave Ψ_B except possibly through nodes already shared with another module, which are absent outside Ψ_A, and symmetrically for Ψ_C.*
5. *Conclude claim 2: when $\Psi_{\underline{BC}} = \emptyset$, any path connecting a node of Ψ_B to a node of Ψ_C must intersect Ψ_A; therefore conditioning on Ψ_A separates the two modules, yielding $\Psi_B \perp \Psi_C \mid \Psi_A$.*

This refinement resolves the observed deficiencies through three structural improvements. First, step 1 explicitly incorporates the closure condition implied by Rule 1, providing a rigorous theoretical foundation for all subsequent deductions. Second, step 3 utilizes this closure property to bound the first exit vertex, excluding inadmissible transitions and replacing the vague eventual reachability argument with rigorous graph-theoretic justification. Third, steps 4 and 5 derive independence by explicitly combining disjointness with the closure property to ensure logical modularity and correctness. By establishing the topological invariants early, StatProver eliminates macro-level trajectory drift.

3.3 Execution of Snippet-Level Self-Correction

Following the establishment of the macro-skeleton, we examine the Snippet-Level Self-Correction module. Even with proper structural guidance, the baseline language model generates the following detailed proof draft, which exhibits subtle micro-level logical leaps in its granular derivations.

Faulty Generated Proof (Pre-Correction):

1. ***Decomposition and Module Properties:*** *Define the disjoint sets $\Psi_{\underline{BC}}$, $\Psi_{\underline{B}}$, and $\Psi_{\underline{C}}$ partitioning the modules outside Ψ_A. The condition $ch(\Psi'_{M^c}) = \emptyset$ states that the set of children of the complement of M is empty. Formally, if $u \notin \Psi'_M$, then u has no children in G. This implies that all edges in G must originate from nodes within a module.*

2. ***Directed Paths from the Overlap:*** *For any $a \in \Psi_{\underline{BC}}$, there exists a directed path to X_B^* and a directed path to X_C^*. By the ancestral closure property derived in step 1, all ancestors of these paths are contained in their respective modules.*
3. ***Proof of Claim 1:*** *Suppose $\Psi_{\underline{BC}} \neq \emptyset$ and let $a \in \Psi_{\underline{BC}}$. Since $a \notin \Psi_A$, we examine the path to X_B^*. The path must eventually exit Ψ_A to reach X_B^* or a module-specific node. In all cases, there exists an edge $a' \to b$ where $a' \in \Psi_{\underline{BC}}$ and b lands in the specific union $(\Psi_{\underline{B}} \cup X_B^*) \cup (\Psi_{\underline{C}} \cup X_C^*)$ or leads directly to it.*
4. ***Structural Consequence of Empty Overlap:*** *If $\Psi_{\underline{BC}} = \emptyset$, then $\Psi'_B \cap \Psi'_C \subseteq \Psi'_A$. The property $ch(\Psi'_{M^c}) = \emptyset$ implies that for any module, there are no edges from the outside. Consequently, any node in $\Psi'_B \setminus \Psi'_A$ can only have parents in Ψ'_B.*
5. ***Proof of Claim 2:*** *We show that Ψ'_A d-separates the modules. Any path from $\Psi'_B \setminus \Psi'_A$ to $\Psi'_C \setminus \Psi'_A$ must pass through the intersection $\Psi'_B \cap \Psi'_C$. Since this intersection is contained in Ψ'_A, every such path is intercepted by the conditioning set, implying $\Psi'_B \perp \Psi'_C \mid \Psi'_A$.*

However, this generated draft exhibits three cascading logical errors. First, in step 1, the model introduces an unjustified strengthening of the closure property, incorrectly extrapolating that all edges in the entire graph must originate from nodes within a module. Second, this misinterpretation propagates into step 3, where the argument concerning directed path traversal lacks proper justification and relies entirely on vague eventual reachability. Third, the conditional independence analysis in step 5 fails to rigorously execute d-separation, completely neglecting the mandatory evaluation of collider versus non-collider structures along active paths.

Operating on the segmented proof draft according to the correction pipeline, the system queries the empirical fault repository. By comparing the generated text against historical failure trajectories, GPT-5.4 retrieves structurally similar error patterns, specifically those involving the misinterpretation of path-blocking boundaries and the improper circumvention of formal d-separation conditions. Guided by these retrieved corrective constraints, the system applies the deferred rectification strategy, yielding the following mathematically rigorous final proof.

Corrected Final Proof (StatProver):

1. ***Decomposition and Module Properties:*** *Define the disjoint sets partitioning the modules. The condition $ch(\Psi'_{M^c}) = \emptyset$ dictates that no node outside a module can have a child inside that module. Formally, if $v \in \Psi'_M$ and $u \to v \in \mathcal{E}$, then $u \in \Psi'_M$. By induction, all ancestors of nodes in Ψ'_M are strictly contained within Ψ'_M.*
2. ***Directed Paths from the Overlap:*** *For any $a \in \Psi_{\underline{BC}}$, valid directed paths to X_B^* and X_C^* exist. Due to the corrected ancestral closure property, transitions from internal nodes to external unassigned nodes in Ψ'_S are strictly prohibited.*
3. ***Proof of Claim 1:*** *We employ a proof by contradiction focused on first edge positioning. Let $a' \to b$ be the first edge leaving the intersection $\Psi_{\underline{BC}}$ along the path to X_B^*. This locks the condition $a' \in \Psi_{\underline{BC}}$ and $b \notin \Psi_{\underline{BC}}$. Since $a' \in \Psi'_B$, its child b must remain in Ψ'_B to avoid violating ancestral closure. Since b cannot belong to Ψ'_S or the conditioning set Ψ'_A, it must transition into the remaining valid subsets, proving $b \in (\Psi_{\underline{B}} \cup X_B^*) \cup (\Psi_{\underline{C}} \cup X_C^*)$.*
4. ***Structural Consequence of Empty Overlap:*** *If $\Psi_{\underline{BC}} = \emptyset$, then $\Psi'_B \cap \Psi'_C \subseteq \Psi'_A$. The rigorous closure property ensures that disjoint submodules cannot share edges without traversing the conditioning set Ψ'_A.*
5. ***Proof of Claim 2:*** *We conduct an explicit analysis of path blocking under conditioning. Consider any undirected path. If the path contains a collider, its parents must reside in the modules containing their respective children by ancestral closure. If the path contains no colliders, any connection between disjoint submodules must pass through their common nodes. In all scenarios, paths are rendered inactive when conditioning on Ψ'_A. This confirms the d-separation principle and yields $\Psi'_B \perp \Psi'_C \mid \Psi'_A$.*

The revised proof successfully rectifies the specific flaws identified in the initial draft. Addressing the first error, step 1 now correctly defines the closure property to explicitly prohibit external nodes from possessing internal children, eliminating the previous overgeneralization. Addressing the second error, step 3 replaces the informal reachability claim with a strict proof by contradiction focused on first edge positioning, ensuring mathematical rigor. Finally, addressing the third error, step 5 conducts a comprehensive analysis covering both collider and non-collider scenarios, confirming the d-separation principle formally. This effectively demonstrates the capability of the snippet-level correction mechanism to enforce absolute rigor in long-chain reasoning.

4 Conclusion

In this paper, we introduce StatProver, an LLM-driven automated proof assistant bridging the capability gap in rigorous statistical derivations. To overcome the challenges of macro-level trajectory drift and micro-level attention decay, we propose a modular pipeline driven by two core mechanisms: a retrieval-augmented framework refinement utilizing a bi-directional max-matching algorithm, and a data-driven snippet-level self-correction based on a massive empirical fault repository. Furthermore, we integrate these mechanisms into a highly flexible HIL workflow. Our evaluations demonstrate that StatProver effectively rectifies reasoning errors, and generates highly accurate mathematical proofs.

References

[1] OpenAI, "Introducing GPT-5.4," https://openai.com/index/introducing-gpt-5-4/, March 2026.

[2] Anthropic, "Claude opus 4.6 system card," Anthropic PBC, Tech. Rep., February 2026. [Online]. Available: https://www.anthropic.com/news/claude-opus-4-6

[3] A. Adcock, A. Srivastava, A. Dubey, A. Jauhri, A. Pande, A. Pandey, A. Sharma, A. Kadian, A. Kumawat, A. Kelsey *et al.*, "The llama 4 herd: Architecture, training, evaluation, and deployment notes," *arXiv preprint arXiv:2601.11659*, 2026.

[4] Google DeepMind, "Gemini 3.1 pro: A smarter model for your most complex tasks," https://blog.google/technology/ai/gemini-3-1-pro-update/, February 2026.

[5] Qwen Team, "Qwen3.6-plus: Towards real world agents," https://qwen.ai/blog?id=qwen3.6, April 2026.

[6] E. Glazer, E. Erdil, T. Besiroglu, D. Chicharro, E. Chen, A. Gunning, C. F. Olsson, J.-S. Denain, A. Ho, E. d. O. Santos *et al.*, "Frontiermath: A benchmark for evaluating advanced mathematical reasoning in ai," *arXiv preprint arXiv:2411.04872*, 2024.

[7] B. Gao, F. Song, Z. Yang, Z. Cai, Y. Miao, Q. Dong, L. Li, C. Ma, L. Chen, R. Xu *et al.*, "Omni-math: A universal olympiad level mathematic benchmark for large language models," *arXiv preprint arXiv:2410.07985*, 2024.

[8] T. H. Trinh, Y. Wu, Q. V. Le, H. He, and T. Luong, "Solving olympiad geometry without human demonstrations," *Nature*, vol. 625, no. 7995, pp. 476–482, 2024.

[9] T. AlphaProof and T. AlphaGeometry, "Ai achieves silver-medal standard solving international 178 mathematical olympiad problems," *DeepMind blog*, vol. 179, p. 45, 2024.

[10] H. Ye, X. He, V. Arak, H. Dong, and G. Song, "Meta context engineering via agentic skill evolution," *arXiv preprint arXiv:2601.21557*, 2026.

[11] T. Tao, "Machine-assisted proof," *Notices of the American Mathematical Society*, vol. 72, no. 1, pp. 6–13, 2025.

[12] T. Feng, T. H. Trinh, G. Bingham, D. Hwang, Y. Chervonyi, J. Jung, J. Lee, C. Pagano, S.-h. Kim, F. Pasqualotto *et al.*, "Towards autonomous mathematics research," *arXiv preprint arXiv:2602.10177*, 2026.

[13] Y. Lu, R. Yang, Y. Zhang, S. Yu, R. Dai, Z. Wang, J. Xiang, S. Gao, X. Ruan, Y. Huang *et al.*, "Stateval: A comprehensive benchmark for large language models in statistics," *arXiv preprint arXiv:2510.09517*, 2025.