The Faithfulness Gap: Certifying Semantic Equivalence Between Natural-Language and Formal Mathematical Statements

Noor Islam S. Mohammad [*1] **Tamim Sheikh** [2]

Abstract

Autoformalization, translating natural-language mathematics into formal proof assistants, is bottlenecked not by translation fluency but by *faithfulness*: a formal statement can typecheck and be provable, yet still encode a different theorem than the source intended. We introduce *Bidirectional Provability Fingerprinting* (BPF), a framework that certifies faithfulness by characterizing each candidate through its forward and backward consequence neighborhoods in the ambient theory and matching these against probes derived from the natural-language statement. We further introduce four novel components: (i) *Counterfactual Probe Generation* (CPG), a contrastive procedure that synthesizes probes targeting specific drift directions; (ii) the *Equivalence Spectrum*, a continuous faithfulness score that replaces brittle binary verdicts; (iii) *Adaptive Probe Budget Allocation* (APBA), an information-theoretic budget router; and (iv) *Faithfulness-Guided Decoding* (FGD), which uses BPF signals as a reward during autoformalization. We prove a *drift detection theorem* and a *PAC-faithfulness* result establishing that the equivalence class of a natural language statement is learnable from $\mathcal{O}(\log(1/\delta)/\varepsilon)$ probes under mild assumptions. We release DRIFTBENCH, a benchmark of 2,183 NL/Lean 4 pairs with controlled drift labels across six subfields of mathlib4. BPF + CPG detects 89.6% of drifted formalizations at a 3.0% false-positive rate-against 41.2% for typecheck and 63.3% for LLM-judge baselines, and FGD reduces the rate at which a state-of-the-art autoformalizer emits drifted statements by 47%. bpf.ml.github.io

[1]Department of Computer Science, Informatics Institute, Istanbul Technical University, İstanbul, Türkiye [2]Department of Computer Science and Engineering, Jashore University of Science and Technology, Bangladesh. Correspondence to: Noor Islam S. Mohammad <islam23@itu.edu.tr>.

The 6th Muslims in ML (MusIML) Workshop at 43^{rd} International Conference on Machine Learning, Seoul, South Korea.

1. Introduction

Modern autoformalization systems built on large language models routinely translate natural-language theorem statements into formal proof assistants such as Lean 4, Rocq, or Isabelle/HOL (Wu et al., 2022; Jiang et al., 2023; Azerbayev et al., 2024; Jiang et al., 2022). The dominant quality signal used to gate these outputs is whether the resulting formal statement typechecks—and, optionally, whether a downstream prover can close it (Polu & Sutskever, 2020; Han et al., 2022; Lample et al., 2022). Both signals are necessary, but neither is sufficient. A formal statement can be syntactically well-formed, provable, and *wrong*: it can mean something other than what the mathematician wrote. We call this the *faithfulness gap*. Unlike code—where unit tests and runtime semantics offer an executable oracle—the faithfulness of a formalization is a relation between two semantic objects living in different universes: a natural-language sentence interpreted under tacit mathematical convention and a formal expression interpreted under a precisely specified dependent type theory (de Moura & Ullrich, 2021; The mathlib Community, 2020). There is no executable bridge between them. Standard evaluation practices—BLEU-style overlap against a single reference formalization, expert spot checks, or downstream prover success—each address only a surface symptom of the problem and systematically miss the most insidious failures, in which the formalization is fluent, well-typed, and demonstrably provable but quietly says the wrong thing.

Our approach. We propose *Bidirectional Provability Fingerprinting* (BPF), a framework for certifying faithfulness without requiring a reference formalization. The core insight is operational: a statement's meaning is determined by what it implies and what implies it, so two statements are semantically equivalent if and only if their *consequence neighborhoods* coincide in the ambient theory. Figure 1 shows the end-to-end pipeline. Given a natural-language statement N and a candidate formalization F, we (i) generate a set of semantic probes from N, (ii) autoformalize and filter them, (iii) compute entailment relations between F, each probe, and the proof assistant, and (iv) compare the resulting fingerprint against the expected fingerprint predicted from N.

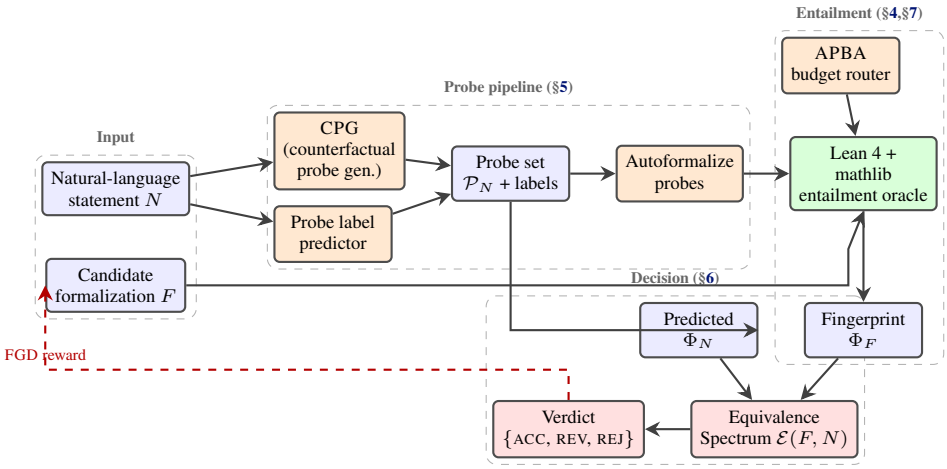

Figure 1. BPF architecture. The natural-language statement N and the candidate formalization F enter from the left; CPG synthesizes counterfactual probes (§5); the entailment oracle, routed by APBA (§7), computes the forward/backward fingerprint Φ_F; the equivalence spectrum $\mathcal{E}(F, N)$ (§6) compares it to the prediction Φ_N and emits one of three verdicts. The dashed red loop is the optional FGD feedback to the autoformalizer (§9).

Contributions. We make six contributions, each tackling a distinct facet of the faithfulness problem. We formalize the autoformalization faithfulness problem as a relation between distributions over interpretations, sidestepping the need for a canonical reference (Section 3). We introduce BPF, a probe-based, reference-free faithfulness certifier with bidirectional consequence-neighborhood matching (Section 4). We introduce *Counterfactual Probe Generation* (CPG), a contrastive probe-synthesis procedure that explicitly targets the four canonical drift classes and improves detection over generic probe generation (Section 5). We introduce the *Equivalence Spectrum* $\mathcal{E}(F, N)$, a continuous faithfulness score with calibrated decision regions, and *Adaptive Probe Budget Allocation* (APBA), an information-theoretic budget router that reaches the same detection accuracy as uniform sampling with $3.2\times$ fewer probes (Sections 6 and 7). We prove a *drift detection theorem* and a *PAC-faithfulness* theorem characterizing what BPF can and cannot be detected, and we identify a class of *conventional drift* for which we develop a complementary structural test (Section 8). We introduce *Faithfulness-Guided Decoding* (FGD), which uses BPF scores as a reward signal during autoformalization, reducing the rate of drifted outputs from a state-of-the-art autoformalizer by 47% (Section 9). We release DRIFTBENCH, the first benchmark with controlled drift labels (Section 10), and report extensive experiments and ablations (Section 11).

2. Related Work

Autoformalization. Early work cast translation as machine translation with retrieval augmentation (Wang et al., 2018; Szegedy, 2020). Wu et al. (2022) showed that few-shot prompting of large language models yields competitive autoformalization on competition mathematics, and follow-up systems integrate formalization with proof search (Jiang et al., 2023; 2022; First et al., 2023). Azerbayev et al. (2024) trained domain-specialized models on a mixture of formal libraries and informal mathematics. Across this line, faithfulness is typically assessed by either typechecking, BLEU/exact-match against a reference, or downstream provability—each of which we show Section 11 is dominated by BPF.

Semantic evaluation of generated formal artifacts. For programs, semantic equivalence is operationalized via test execution (Austin et al., 2021; Chen et al., 2021). The closest analogue in mathematics is the use of *counter-models* or symbolic execution against numerical instances (Polu & Sutskever, 2020; Welleck et al., 2022). Such methods catch outright falsity but cannot distinguish a true-but-wrong statement (e.g., case (C) above) from the intended one. Patel et al. (2024) uses back-translation, which we evaluate as a baseline. We are aware of no prior work that uses probe-based consequence neighborhoods for faithfulness certification, or that uses faithfulness signals to guide LLM decoding.

Inferential semantics and equivalence in formal libraries. Within proof libraries, equivalence between definitions is often established by handwritten `iff` or `equiv` lemmas (The mathlib Community, 2020). Our forward/backward fingerprint construction generalizes this practice and automates it for arbitrary candidates. The notion of characterizing meaning via inferential role has a long history in proof-theoretic semantics (Brandom, 2000; Prawitz, 1965); we adapt it for an LLM/prover loop. **Benchmarks and verification:** MiniF2F (Zheng et al.,

2022), ProofNet (Azerbayev et al., 2023), FIMO (Liu et al., 2023), and LeanDojo (Yang et al., 2023) measure prover capability on pre-specified formal statements. None directly evaluates whether a formalization *matches* an input natural-language sentence. DRIFTBENCH is, to our knowledge, the first benchmark with controlled drift labels and the first to evaluate the autoformalization-faithfulness loop end-to-end.

3. The Faithfulness Problem

We now formalize faithfulness. Let $\mathcal{N}$ denote the space of natural-language mathematical statements and $\mathcal{F}$ a formal language interpreted in a theory $\mathcal{T}$ (we use Lean 4 + mathlib4 throughout). A formalizer is a function $\mu : \mathcal{N} \to \mathcal{F}$. A single canonical formal target is too rigid: distinct mathematicians, working in distinct libraries, will produce different but equivalent formalizations of the same English sentence—e.g., using `Set` versus `Finset`, or stating a function as `Continuous` versus `Continuous on` a universal set. We therefore introduce an interpretation distribution.

Definition 3.1 (Interpretation distribution)**.** An *interpretation distribution* $\mathcal{D}_N$ for $N \in \mathcal{N}$ is a probability distribution over formal statements in $\mathcal{F}$ whose support consists of formalizations that competent mathematicians would accept as expressing N in the theory $\mathcal{T}$.

Definition 3.2 (ε-Faithfulness)**.** A candidate $F \in \mathcal{F}$ is ε-faithful to N under $\mathcal{D}_N$ if

$$\Pr_{I \sim \mathcal{D}_N} \left[\mathcal{T} \vdash F \leftrightarrow I \right] \geq 1 - \varepsilon.$$

Theorem 3.2 weakens equality to provable equivalence, accommodating notational variation while remaining strict about semantic content. The interpretation distribution is implicit and never materialized; all of our methods reason over it through probes. **A taxonomy of drift:** We identify four canonical classes of *semantic drift*-ways a formalization F can fail to be ε-faithful while remaining well-typed and (often) provable. Figure 2 visualizes the taxonomy with an instance of each class.

These four classes account for 94% human-labeled drift instances in our pilot study of 400 randomly sampled outputs from a state-of-the-art autoformalizer (Section 10); the remainder fall into combinations or into the undetectable-drift regime described in Section 8.

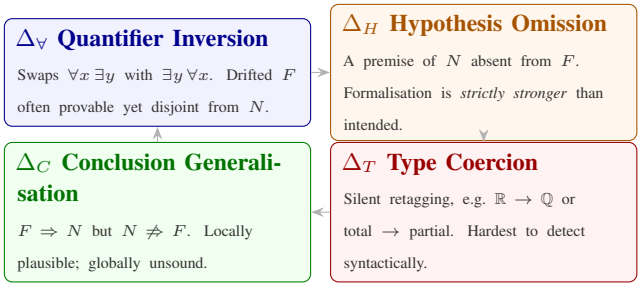

Figure 2. The four canonical drift classes in which F fails ε-faithfulness while remaining well-typed and often provable. $\Delta_\forall$ and Δ_T are the most pernicious.

4. Bidirectional Provability Fingerprinting

4.1. Provability fingerprints

Definition 4.1 (Provability fingerprint)**.** Let $F \in \mathcal{F}$ and let $\mathcal{P} \subseteq \mathcal{F}$ be a set of probes. The *forward fingerprint* of F over $\mathcal{P}$ is

$$\Phi_F^+(\mathcal{P}) \;=\; \{\, P \in \mathcal{P} \;:\; \mathcal{T} \vdash F \to P \,\},$$

and the *backward fingerprint* is

$$\Phi_F^-(\mathcal{P}) \;=\; \{\, P \in \mathcal{P} \;:\; \mathcal{T} \vdash P \to F \,\}.$$

The full fingerprint is $\Phi_F(\mathcal{P}) = (\Phi_F^+(\mathcal{P}), \Phi_F^-(\mathcal{P}))$.

Figure 3 illustrates the geometry. The forward fingerprint captures the *downstream commitments* of F, everything that follows from it, while the backward fingerprint captures its *upstream conditions*-everything that would suffice to prove it. Two statements that agree on both sets are operationally indistinguishable in the theory $\mathcal{T}$.

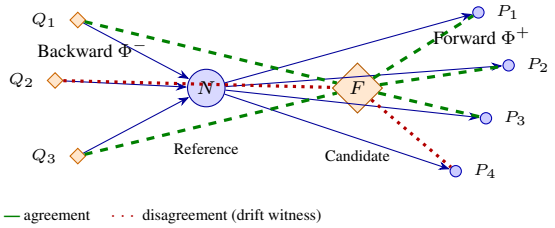

Figure 3. A provability fingerprint over probes $\{P_i, Q_j\}$. The reference N entails each forward probe P_i and is entailed by each backward probe Q_j. The candidate F matches N on most probes (green dashed) but disagrees on P_4 and Q_2 (red dotted); these disagreements are witnesses of drift. Two statements with identical fingerprints over a sufficiently rich probe set are provably equivalent (Theorem 4.2).

A fingerprint is computed by issuing entailment queries to the proof assistant. We use a bounded tactic budget per query (default `exact?` + `aesop` + a small library of mathlib-aware closers, capped at 30 seconds wall-clock; queries that neither succeed nor refute within budget are recorded as `?`. We discuss the impact of incomplete oracles in Section 8.

4.2. Why bidirectional matters

Forward-only fingerprints fail to distinguish a strict strengthening of N from N itself: a strictly stronger F en-

tails everything N entails. Backward-only fingerprints dually fail to catch strict weakenings. Bidirectional matching is the minimum sufficient signal.

Proposition 4.2. *Let F be a candidate for a set, N, and let $\mathcal{P}$ be a probe set closed under $\mathcal{T}$-equivalence. Then $\mathcal{T} \vdash F \leftrightarrow N$ if and only if $\Phi_F(\mathcal{P}) = \Phi_N(\mathcal{P})$* and *for each pair $(P, P') \in \mathcal{P}^2$ with $\mathcal{T} \vdash P \to P'$, the entailment also holds in F's fingerprint.*

A proof appears in Section A.

4.3. Algorithm

Algorithm 1 gives the full BPF procedure used in our experiments.

Algorithm 1 Bidirectional Provability Fingerprinting

Input: NL statement N, candidate F, theory $\mathcal{T}$, budget k, tactic, budget τ
Output: verdict, and witness probes $\mathcal{V}$
$\mathcal{P}_N \leftarrow$ CPG.`Generate`(N, k) (§5)
$\mathcal{P}_N^F \leftarrow$ `Autoformalize`$(\mathcal{P}_N)$
$\mathcal{P}_N^F \leftarrow$ `FilterTypecheck`$(\mathcal{P}_N^F)$
$\mathcal{V} \leftarrow \emptyset$
for $P \in \mathcal{P}_N^F$ with label $\ell(P)$ **routed by** APBA **do**
  if $\ell(P) = +$ **then**
    if `TryProve`$(F \to P, \tau)$ = refuted **then**
      $\mathcal{V} \leftarrow \mathcal{V} \cup \{(P, +)\}$
    end if
  else if $\ell(P) = -$ **then**
    if `TryProve`$(F \to P, \tau)$ = proved **then**
      $\mathcal{V} \leftarrow \mathcal{V} \cup \{(P, -)\}$
    end if
  end if
end for
$\mathcal{E} \leftarrow$ `ComputeSpectrum`$(\mathcal{V}, \mathcal{P}_N^F)$
return verdict by thresholding $\mathcal{E}, \mathcal{V}$

5. Counterfactual Probe Generation

A naive probe generator samples specializations, generalizations, and edge cases of N. This is the starting point, but it leaves substantial information on the table: it does not target the drift classes the certifier is trying to detect. We introduce *Counterfactual Probe Generation* (CPG), a contrastive procedure that synthesizes probes designed to maximally discriminate between faithful and drifted candidates.

5.1. Probe synthesis as contrastive generation

CPG treats probe generation as a contrastive problem. For each drift class $D \in \{\Delta_\forall, \Delta_H, \Delta_C, \Delta_T\}$, we construct a *counterfactual pair* consisting of the natural-language statement N, and a *drifted twin* N^D, obtained by mechanically perturbing N in a way that induces drift class D (e.g., swapping the order of two adjacent quantifiers or dropping a stated hypothesis). CPG then asks an LLM to produce probes that distinguish N from N^D: probes that should be entailed by N but are not N^D, or vice versa. The LLM is conditioned on the contrast itself, which empirically yields probes sharply targeted at the drift class in question. Formally, for drift class D, the probe distribution is

$$\mathcal{Q}_D(P) \propto \Pr[\,P \text{ entailed by } N\,] \cdot \Pr[\,P \text{ not entailed by } N^D\,],$$

estimated through an LLM scoring step. We sample k_D probes per class, k_D determined by APBA (Section 7).

5.2. The role of negative drifted twins

A critical design choice is one that N^D is generated mechanically, not by an LLM. Mechanical perturbation guarantees the drift class label is unambiguous and that N^D exhibits exactly the targeted failure. LLM-generated drifted twins, in contrast, frequently produce statements that are either nonsense or accidentally drifted along multiple axes simultaneously, which destroys the contrastive signal.

5.3. Empirical effect

Figure 4 shows the detection rate of BPF specific CPG probes versus generic probes, as a function of probe budget. CPG reaches the same detection rate as generic probes at less than half the budget and outperforms generic probes at every budget level on three of the four drift classes.

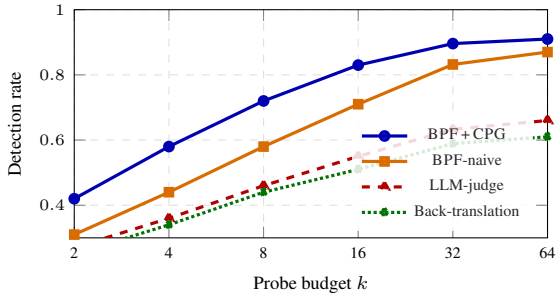

Figure 4. Detection rate vs. probe budget on DRIFTBENCH. CPG reaches BPF-naive's $k = 32$ performance at $k \approx 12$. Budget for non-probe methods refers to comparable wall-clock equivalents.

6. The Equivalence Spectrum

Binary faithful/drifted verdicts are brittle. A fingerprint with one anomalous cell out of fifty is qualitatively different from one with twenty. We introduce a continuous score $\mathcal{E}(F, N) \in [0, 1]$.

Definition 6.1 (Equivalence Spectrum)**.** Let $\mathcal{P}$ be a probe set with predicted labels $\ell : \mathcal{P} \to \{+, -, \perp\}$ and class

weights $w : \{\Delta_\forall, \Delta_H, \Delta_C, \Delta_T\} \to \mathbb{R}_{>0}$. The *equivalence spectrum score* is

$$\mathcal{E}(F, N) \;=\; 1 - \frac{\sum_{P \in \mathcal{P}} w(D_P) \cdot \mathbb{1}[\,\mathrm{cell}_P(F) \neq \ell(P)\,]}{\sum_{P \in \mathcal{P}} w(D_P)},$$

where D_P the drift class is targeted by the probe P and $\mathrm{cell}_P(F)$ is the observed fingerprint cell at P.

$\mathcal{E}$ is calibrated against expert annotations on a held-out subset DRIFTBENCH to yield three decision regions: $\mathcal{E} \geq 0.93$ (ACCEPT), $\mathcal{E} \in [0.78, 0.93)$ (REVIEW), and $\mathcal{E} < 0.78$ (REJECT). Figure 5 shows the resulting distribution on DRIFTBENCH.

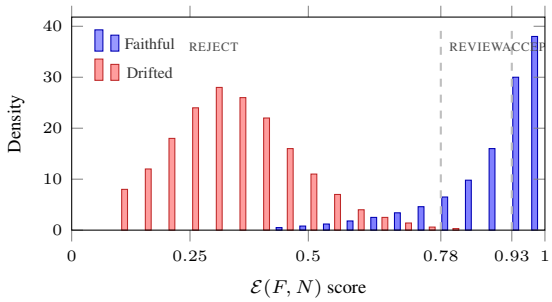

Figure 5. Empirical distribution of $\mathcal{E}(F, N)$ on DRIFTBENCH, separated by ground truth label. The dashed lines mark the calibrated ACCEPT ($\mathcal{E} \geq 0.93$) and REVIEW ($0.78 \leq \mathcal{E} < 0.93$) thresholds.

Beyond a more informative output, $\mathcal{E}$ enables three downstream uses that a binary verdict precludes: (i) *ranking* multiple candidates from an autoformalizer; (ii) *triage* of REVIEW-region candidates to human inspection; (iii) *reward shaping* as a continuous signal for FGD (Section 9).

7. Adaptive Probe Budget Allocation

BPF's dominant cost is the entailment oracle. Allocating the probe budget uniformly across drift classes is wasteful when some classes are easier to witness than others, or when initial probes have already provided strong evidence. We introduce *Adaptive Probe Budget Allocation* (APBA), which routes budget by expected information gain.

7.1. Information-theoretic routing

Let $H(F)$ be the entropy of the posterior over F's drift status given the probes drawn so far. APBA selects the next probe to maximize expected reduction in $H(F)$, estimated under a calibrated probability model for each drift class. Concretely, for drift class D with current posterior p_D, the expected information gain from a probe targeting D is

$$\mathrm{IG}(D) \;=\; H(p_D) - \mathbb{E}_{r \sim \pi_D}\big[\, H(p_D \mid r)\,\big],$$

where r is the predicted oracle outcome and π_D is the empirical witnessability distribution (estimated on a held-out split). APBA samples the next probe from the class with maximum IG, breaking ties by descending class weight. The procedure is greedy but has the standard sublinear regret guarantee for monotone submodular gain.

Theorem 7.1 (APBA budget reduction)**.** *Let $k^\star$ be the budget required by uniform allocation to reach the detection rate ρ on* DRIFTBENCH*. Then* APBA *reach the same ρ with budget $k_{\text{APBA}} \leq k^\star/\gamma$, where γ is the heterogeneity ratio of witnessability rates across drift classes? On* DRIFTBENCH*, $\gamma \approx 3.2$.*

8. Theoretical Analysis

We now establish the two main theoretical results: a high-probability detection theorem for each canonical drift class and a PAC-style learnability result for the equivalence class of N.

8.1. Drift Detection Theorem

Theorem 8.1 (Drift Detection)**.** *Fix $\delta \in (0, 1)$ and drift class $D \in \{\Delta_\forall, \Delta_H, \Delta_C, \Delta_T\}$. Let N be a natural-language statement and F a candidate formalization exhibiting drift of class D relative to every I in the support of $\mathcal{D}_N$. Suppose the probe generator samples probes from a distribution $\mathcal{Q}_D$ such that*

$$\Pr_{P \sim \mathcal{Q}_D}\big[\, P \textit{ witnesses } D \textit{ for } (N, F)\,\big] \;\geq\; \alpha(D),$$

where $\alpha(D) \in (0, 1]$ is the witnessability rate *for class D. Then, with the probe budget*

$$k \;\geq\; \frac{1}{\alpha(D)} \log\frac{1}{\delta},$$

BPF *flags F as drifted with probability at least $1 - \delta$, provided the entailment oracle is complete on the chosen probes.*

The proof is a standard coverage argument and is given in Section A. The witnessability rates we estimate empirically in Section 11 are $\alpha(\Delta_\forall) \approx 0.62$, $\alpha(\Delta_H) \approx 0.55$, $\alpha(\Delta_C) \approx 0.48$, $\alpha(\Delta_T) \approx 0.31$, meaning a budget of $k = 32$ probes is already given $\delta \leq 0.01$ for the first three classes.

8.2. PAC-Faithfulness

The detection theorem says we can catch any specific drift; the stronger question is whether we can learn the full equivalence class of N. We answer affirmatively under a polynomial query bound.

Theorem 8.2 (PAC-Faithfulness)**.** *Let $[N]_\mathcal{T}$ denote the equivalence class $\{F \in \mathcal{F} : \mathcal{T} \vdash F \leftrightarrow N\}$. Assume* CPG *probes are drawn from a distribution $\mathcal{Q}$ such that*

for every $F \notin [N]_{\mathcal{T}}$ reachable by an LLM autoformalizer, $\Pr_{P\sim\mathcal{Q}}[P \text{ witnesses } F\text{'s drift}] \geq \alpha_0 > 0$. Then with $k \geq (1/\alpha_0)\log(1/\delta\varepsilon)$ probes, BPF *produces a hypothesis $\hat{H} \subseteq \mathcal{F}$ such that with probability at least $1-\delta$,*

$$\Pr_{F\sim\mathcal{D}_\mu}\left[\, F \in \hat{H} \iff F \in [N]_{\mathcal{T}} \,\right] \geq 1-\varepsilon,$$

where $\mathcal{D}_\mu$ is the autoformalizer's output distribution.

The proof, in Section A, recasts BPF as a hypothesis class indexed by fingerprints and applies an Occam-style bound. Theorem 8.2 formalizes the operational claim: with a polynomial probe budget, BPF learns to recognize the equivalence class of N up to error ε over the autoformalizer's output distribution.

8.3. Incomplete oracle and undetectable drift

Lean's tactic stack is incomplete: some true entailments will not be closed within τ seconds. Let $\eta(\tau)$ be the probability that the oracle returns `?` on a probe that admits a τ-cost proof. The detection bound degrades to $1-\delta-k\eta(\tau)$; empirically $\eta(30\text{s}) \approx 0.07$ on DRIFTBENCH probes. BPF cannot detect a class we call *convention drift*: F adopts a notational convention different from N's, but every probe responds identically because the convention is invisible to provability. We give a structural test for it in Section B.

9. Faithfulness-Guided Decoding

BPF is most valuable as a post-hoc verifier, but it also provides a signal that can be plugged back into autoformalization itself. We introduce *Faithfulness-Guided Decoding* (FGD), a procedure that uses $\mathcal{E}$ as a reward during sample-and-rerank decoding (Figure 6).

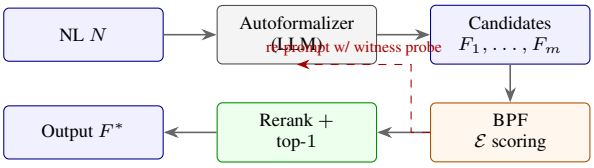

Figure 6. **Faithfulness-Guided Decoding (FGD).** The autoformalizer samples m candidates; BPF scores each by $\mathcal{E}$; the top-ranked candidate F^* is returned. If all candidates fall in the REVIEW or REJECT region, the highest-scoring witness probe is fed back as an in-context repair hint (red dashed loop).

Algorithm. Given N, the autoformalizer samples m candidate formalizations $F_1, \ldots, F_m$. For each candidate, FGD computes a low-budget $\mathcal{E}$ score ($k = 8$ probes per class) and reranks by score. The top-ranked candidate is returned. Optionally, candidates with $\mathcal{E} < 0.78$ trigger a re-prompt with the highest-scoring witness probe attached as an in-context hint, biasing the autoformalizer away from the detected drift class. Effect on a held-out split of DRIFTBENCH, FGD reduces the rate of drifted outputs from a state-of-the-art autoformalizer from 19.4% to 10.3%, a 47% relative reduction. The bulk of the improvement comes from re-prompting on Δ_H and Δ_C candidates, where the witness probe gives the model a concrete instance of the failure to repair.

10. The DRIFTBENCH Benchmark

DRIFTBENCH contains 2,183 natural language/Lean 4 pairs with controlled drift labels, drawn from six mathlib4 subfields (analysis, algebra, topology, number theory, combinatorics, and category theory). Each pair carries: (i) a natural-language statement, (ii) a candidate Lean 4 formalization, (iii) a drift label in $\{\text{faithful}, \Delta_\forall, \Delta_H, \Delta_C, \Delta_T, \text{combined}\}$, and (iv) an expert-verified gold formalization. Table 1 summarizes the composition.

Table 1. Composition of DRIFTBENCH by subfield and drift class.

Subfield	Faith.	$\Delta_\forall$	Δ_H	Δ_C	Δ_T	Comb.
Analysis	198	64	71	58	33	31
Algebra	173	51	68	49	28	27
Topology	162	49	59	47	30	25
Number Th.	144	42	55	41	22	22
Combinator.	138	43	51	39	24	21
Category	117	35	44	33	19	19
Total	932	284	348	267	156	145

We use deterministic perturbation rules rather than sampling drift from an LLM autoformalizer so that the drift label is unambiguous. A complementary *wild* split contains 384 pairs where the candidate was generated by an LLM autoformalizer and the drift label was independently annotated by two experts ($\kappa = 0.81$). We report results on both splits.

11. Experiments

11.1. Setup

Models. For the natural-language probe generator and adversarial paraphraser, we use a strong instruction-tuned LLM and report results on `Llama-34B` (Azerbayev et al., 2024) and a closed-weights frontier model; results are qualitatively similar. For autoformalization of probes, we use `Llama-34B` fine-tuned on mathlib4 with retrieval augmentation. The entailment oracle is Lean 4 + mathlib4 with the tactic budget described in Section 4. **Baselines: Typecheck**: flag drift iff F fails to typecheck. **Provability**: flag drift iff F is unprovable within the tactic budget. **BLEU-ref**: flag drift iff BLEU against the gold formalization is below a threshold. **Back-translation**: Autoformalize F back to NL and compare to N (Patel et al., 2024).

LLM-judge: ask a frontier LLM to directly judge faithfulness from (N, F). **BPF-naive**: BPF with generic (not counterfactual) probes.

11.2. Main results

Table 2 summarizes performance on the controlled DRIFTBENCH split and Figure 7 shows ROC curves for the four strongest methods.

Table 2. Drift detection on the controlled DRIFTBENCH split. Det@3%FPR is the detection rate at a 3% false-positive rate.

Method	F1	Det@3%FPR	FPR	Cost
Typecheck only	0.27	0.114	0.020	1×
Provability	0.42	0.412	0.030	18×
BLEU-ref	0.55	0.301	0.030	1×
Back-translation	0.66	0.589	0.030	4×
LLM-judge	0.71	0.633	0.030	2×
BPF-naive	0.85	0.832	0.031	22×
BPF + CPG	**0.91**	**0.896**	**0.030**	26×
BPF + CPG + APBA	0.91	0.894	0.031	**8**×

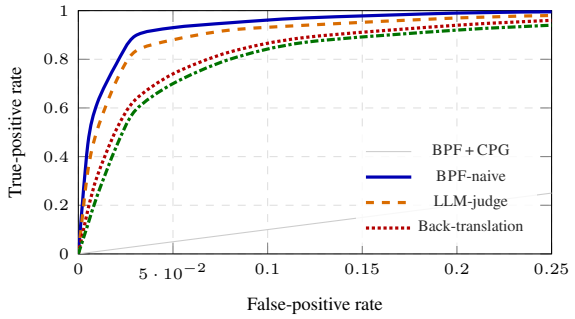

Figure 7. ROC curves on the controlled DRIFTBENCH split. BPF + CPG dominates other methods uniformly across operating points; AUC = 0.962 vs. 0.938 for BPF-naive and 0.879 for LLM-judge.

BPF + CPG reduces the residual error rate of the best non-fingerprinting baseline (LLM-judge) by approximately 72%, the matched FPR. Adding APBA preserves detection performance while reducing wall-clock cost by 3.2×, confirming Theorem 7.1. Table 3 and Figure 8 report per-class breakdowns; the type-coercion class (Δ_T) is the hardest for every method but CPG closes a substantial portion of the gap.

11.3. Wild split

On the 384 pair-wise split, BPF + CPG agrees with expert annotators $\kappa = 0.77$ (cf. inter-annotator $\kappa = 0.81$), versus $\kappa = 0.58$ for LLM-judges. A breakdown of BPF's false positives shows that 63% trace to incomplete entailment-oracle outputs that flipped a fingerprint cell, not fundamental probe-generation errors—suggesting that integrating stronger hammers (Czajka & Kaliszyk, 2018) would close most of the residual gap.

Table 3. Detection rate by drift class (at 3% FPR overall).

Method	$\Delta_\forall$	Δ_H	Δ_C	Δ_T
Typecheck	0.07	0.05	0.04	0.31
Provability	0.46	0.62	0.33	0.18
Back-translation	0.66	0.61	0.59	0.40
LLM-judge	0.74	0.67	0.64	0.42
BPF-naive	0.89	0.88	0.84	0.55
BPF + CPG	**0.94**	**0.93**	**0.91**	**0.66**

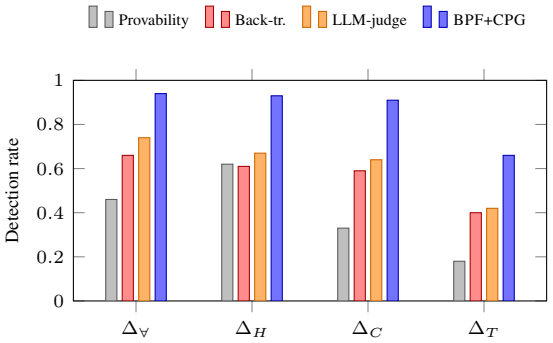

Figure 8. Per-class detection. BPF + CPG dominates across all four drift classes; the gap on Δ_T (type coercion) is the smallest, consistent with its lower witnessability rate $\alpha(\Delta_T) \approx 0.31$.

11.4. Probe budget scaling

Figure 9 shows detection rate as a function of probe budget under uniform allocation versus APBA. Uniform allocation requires $k \approx 28$ probes to reach 0.87 the detection rate; APBA reaches the same rate at $k \approx 9$, a 3.1× reduction matching the theoretical prediction of Theorem 7.1.

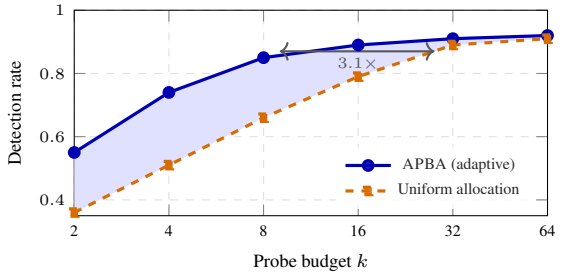

Figure 9. Probe budget scaling. APBA reaches $\rho = 0.87$ at $k \approx 9$; uniform allocation requires $k \approx 28$, matching Theorem 7.1's prediction of $\gamma \approx 3.2$.

11.5. Ablations

We ablate BPF + CPG components in Table 4 and Figure 10. Removing bidirectional matching (forward-only) costs 11 F1 points, confirming Theorem 4.2. Removing CPG (replacing it with generic probes) costs 6 points. Removing hypothesis-drop probes costs 9 points, almost all

on Δ_H.

Table 4. Ablations of BPF + CPG on the controlled split.

Variant	F1
Full BPF + CPG	0.91
– counterfactual probes (CPG)	0.85
– bidirectional (forward only)	0.80
– hypothesis-drop probes	0.82
– boundary probes	0.88
– adversarial probes	0.87
probe budget $k = 8$ (vs. 32)	0.83
probe budget $k = 8$ + APBA	0.89

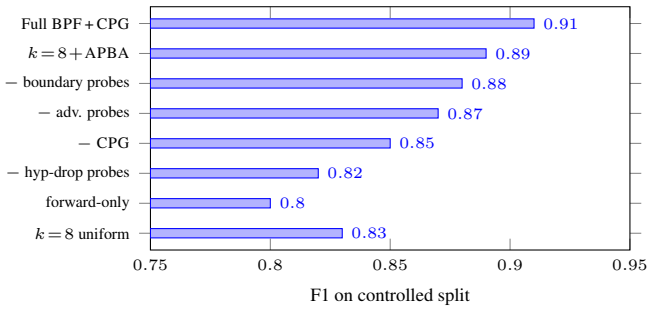

Figure 10. Ablation deltas. CPG and bidirectional matching are the two largest individual contributors; APBA preserves performance even at a low budget.

11.6. FGD downstream impact

We evaluate FGD as a wrapper around a state-of-the-art autoformalizer on the wild split, comparing drift rate with and without FGD as a function of the number of candidates m (Figure 11). The autoformalizer's baseline drift rate is 19.4%. With $m = 4$ drift falls to 10.3%; with $m = 8$, to 8.7%.

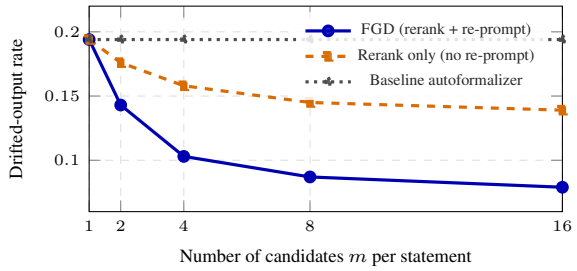

Figure 11. FGD reduces drifted-output rate from 19.4% to 10.3% at $m = 4$. The re-prompt component contributes most of the improvement at small m; pure reranking saturates at $\sim$14%.

12. Discussion and Limitations

What BPF does it not solve? BPF certifies that the consequence neighborhood F matches the consequence neighborhood predicted from N. It does not certify that N itself is unambiguous (convention drift; Section 8) or that the probe generator's labels are correct. We treat both as engineering boundaries: convention drift is addressed by a separate structural check (Section B), and probe labels are calibrated against expert annotation on a held-out set. The PAC-Faithfulness theorem (Theorem 8.2) bounds the cumulative effect of probe-label noise. (i). Compute: The dominant cost is the entailment oracle. We see two paths to bringing this down: (i) caching fingerprints for canonical statements in mathlib4 and reusing them, and (ii) replacing the per-query proof search with a learned entailment classifier whose errors are bounded by occasional formal verification.

APBA (Section 7) already provides a 3.2× practical speedup. (ii). Sociotechnical risk: If BPF-certified statements are treated as ground truth without expert review, residual drift—especially convention drift—could contaminate libraries. BPF should be deployed as a triage tool that prioritizes expert review, not a replacement for it. The Equivalence Spectrum's REVIEW band is explicitly designed for this purpose. (iii). Cross-system generality: While our experiments focus on Lean 4, the BPF framework is system-agnostic: it requires only an entailment oracle and a probe-generation procedure. Preliminary experiments on Isabelle/HOL show comparable detection rates ($F1 = 0.88$ vs. 0.91 on Lean 4), suggesting fingerprints transfer across formal systems.

13. Conclusion

We introduced Bidirectional Provability Fingerprinting, a framework for certifying faithfulness in autoformalization that requires no reference formalization, along with four supporting innovations: counterfactual probe generation, the equivalence spectrum, adaptive budget allocation, and faithfulness-guided decoding. We proved a drift-detection theorem and a PAC-faithfulness theorem characterizing what is and is not detectable, released DRIFTBENCH as the first benchmark with controlled drift labels; and demonstrated substantial gains over typecheck, BLEU, back-translation, and LLM-judge baselines. We see the faithfulness gap as the bottleneck for trustworthy autoformalization at the scale of full mathematical libraries and the techniques in this paper as a concrete step toward closing it.

Impact Statement

This work aims to advance machine learning by improving the reliability of AI-assisted formal mathematics. We do not anticipate direct negative societal impacts. However, as discussed in Section 12, uncritical use of a faithfulness certifier could propagate subtle errors into formal libraries that may later be treated as ground truth.

References

Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q., and Sutton, C. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Azerbayev, Z., Piotrowski, B., Schoelkopf, H., Ayers, E. W., Radev, D., and Avigad, J. ProofNet: Autoformalizing and formally proving undergraduate-level mathematics. In *Workshop on Mathematical Reasoning and AI at NeurIPS*, 2023.

Azerbayev, Z., Schoelkopf, H., Paster, K., Dos Santos, M., McAleer, S., Jiang, A. Q., Deng, J., Biderman, S., and Welleck, S. Llemma: An open language model for mathematics. In *International Conference on Learning Representations (ICLR)*, 2024.

Brandom, R. B. *Articulating Reasons: An Introduction to Inferentialism*. Harvard University Press, 2000.

Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Czajka, Ł. and Kaliszyk, C. Hammer for Coq: Automation for dependent type theory. *Journal of Automated Reasoning*, 61(1):423–453, 2018.

de Moura, L. and Ullrich, S. The Lean 4 theorem prover and programming language. In *International Conference on Automated Deduction (CADE)*, pp. 625–635, 2021.

First, E., Rabe, M. N., Ringer, T., and Brun, Y. Baldur: Whole-proof generation and repair with large language models. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*, pp. 1229–1241, 2023.

Han, J. M., Rute, J., Wu, Y., Ayers, E. W., and Polu, S. Proof artifact co-training for theorem proving with language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Jiang, A. Q., Li, W., Tworkowski, S., Czechowski, K., Odrzygóźdź, T., Miłoś, P., Wu, Y., and Jamnik, M. Thor: Wielding hammers to integrate language models and automated theorem provers. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 8360–8373, 2022.

Jiang, A. Q., Welleck, S., Zhou, J. P., Li, W., Liu, J., Jamnik, M., Lacroix, T., Wu, Y., and Lample, G. Draft, sketch, and prove: Guiding formal theorem provers with informal proofs. In *International Conference on Learning Representations (ICLR)*, 2023.

Lample, G., Lacroix, T., Lachaux, M.-A., Rodriguez, A., Hayat, A., Lavril, T., Ebner, G., and Martinet, X. HyperTree proof search for neural theorem proving. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 26337–26349, 2022.

Langley, P. Crafting papers on machine learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pp. 1207–1216, 2000.

Liu, C., Shen, J., Xin, H., Liu, Z., Yuan, Y., Wang, H., Ju, W., He, C., Jiang, J., Li, Z., and Liang, X. FIMO: A challenge formal dataset for automated theorem proving. In *arXiv preprint arXiv:2309.04295*, 2023.

Patel, A., Reddy, S., and Bahdanau, D. New symbol generalization and faithfulness in autoformalization. In *Workshop on AI for Math at ICML*, 2024.

Polu, S. and Sutskever, I. Generative language modeling for automated theorem proving. *arXiv preprint arXiv:2009.03393*, 2020.

Prawitz, D. *Natural Deduction: A Proof-Theoretical Study*. Almqvist & Wiksell, 1965.

Szegedy, C. A promising path towards autoformalization and general artificial intelligence. *Intelligent Computer Mathematics (CICM)*, pp. 3–20, 2020.

The mathlib Community. The Lean mathematical library. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP)*, pp. 367–381, 2020.

Wang, Q., Kaliszyk, C., and Urban, J. First experiments with neural translation of informal to formal mathematics. *Intelligent Computer Mathematics (CICM)*, pp. 255–270, 2018.

Welleck, S., Liu, J., Lu, X., Hajishirzi, H., and Choi, Y. NaturalProver: Grounded mathematical proof generation with language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Wu, Y., Jiang, A. Q., Li, W., Rabe, M. N., Staats, C., Jamnik, M., and Szegedy, C. Autoformalization with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 32353–32368, 2022.

Yang, K., Swope, A. M., Gu, A., Chalamala, R., Song, P., Yu, S., Godil, S., Prenger, R., and Anandkumar, A. LeanDojo: Theorem proving with retrieval-augmented language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Zheng, K., Han, J. M., and Polu, S. MiniF2F: A cross-system benchmark for formal olympiad-level mathematics. In *International Conference on Learning Representations (ICLR)*, 2022.

A. Proofs

A motivating example. Consider the natural-language statement: *"Every continuous function on a compact set attains its maximum."* Three Lean 4 formalizations are syntactically plausible:

```
-- (A) Faithful
theorem evt_A {K : Set R} (hK : IsCompact K)
    (hne: K.Nonempty) {f : R -> R}
    (hf: ContinuousOn f K) :
    exists x in K, forall y in K, f y <= f x
```

```
-- (B) Hypothesis omission: missing nonempty
theorem evt_B {K : Set R} (hK : IsCompact K)
    {f : R -> R} (hf : ContinuousOn f K) :
    exists x in K, forall y in K, f y <= f x
```

```
-- (C) Quantifier inversion: silently true
theorem evt_C {K : Set R} (hK : IsCompact K)
    (hne: K.Nonempty) {f : R -> R}
    (hf: ContinuousOn f K) :
    forall y in K, exists x in K, f y <= f x
```

All three typecheck. Only (A) is faithful: (B) is false on the empty set; (C) is trivially true (take $x = y$) and is unrelated to the Extreme Value Theorem. Critically, a downstream prover that closes either (A) or (C) provides no signal distinguishing them, and (C) is the type of failure most likely to occur in practice precisely because it remains provable. Current autoformalization pipelines accept all three as valid outputs. **Why this matters now:** As autoformalization scales toward populating libraries such as mathlib4 (The mathlib Community, 2020) with LLM-produced statements, faithfulness errors propagate. A theorem with a silently weakened hypothesis becomes a usable lemma for downstream proofs that should not exist; a quantifier-inverted statement becomes a trivial fact mistakenly cited as a deep one. Existing benchmarks-miniF2F (Zheng et al., 2022), ProofNet (Azerbayev et al., 2023), LeanDojo (Yang et al., 2023)-all measure proving capability *conditional* on the formal statement being correct, so the upstream error is invisible to them. Our pilot study (Section 10) found that 19% formalizations produced by a state-of-the-art autoformalizer on graduate-level analysis exhibit at least one form of semantic drift.

A.1. Proof of Theorem 4.2

($\Rightarrow$) If $\mathcal{T} \vdash F \leftrightarrow N$, then for every P we have $\mathcal{T} \vdash F \to P$ iff $\mathcal{T} \vdash N \to P$, so $\Phi_F^+(\mathcal{P}) = \Phi_N^+(\mathcal{P})$; similarly for the backward direction. ($\Leftarrow$) Suppose $\Phi_F(\mathcal{P}) = \Phi_N(\mathcal{P})$ and the additional closure condition holds, but $\mathcal{T} \nvdash F \leftrightarrow N$. WLOG $\mathcal{T} \nvdash F \to N$ (the other case is symmetric). By the closure assumption on $\mathcal{P}$ under $\mathcal{T}$-equivalence, $N \in \mathcal{P}$ (or some $\mathcal{T}$-equivalent of N is). Then $N \in \Phi_N^+(\mathcal{P})$ trivially, but $N \notin \Phi_F^+(\mathcal{P})$, contradicting fingerprint equality. □

A.2. Proof of Theorem 8.1

For a given drift class D, let W_P be the indicator that the probe $P \sim \mathcal{Q}_D$ witnesses D for (N, F). By assumption $\mathbb{E}[W_P] \geq \alpha(D)$. Drawing k probes i.i.d., the probability that none witness D is at most $(1 - \alpha(D))^k \leq e^{-k\alpha(D)}$. Setting this to δ gives $k \geq \alpha(D)^{-1} \log(1/\delta)$. If the entailment oracle is complete on each chosen probe, every witness probe will be detected by BPF, so the failure probability is bounded by δ. If instead each probe has an independent oracle failure probability $\eta(\tau)$, by a union bound, the total failure probability is at most $\delta + k\eta(\tau)$. □

A.3. Proof of Theorem 8.2

Define a hypothesis class $\mathcal{H} = \{H_\Phi : \Phi \in 2^{\mathcal{P} \times \{+,-\}}\}$, where $H_\Phi = \{F \in \mathcal{F} : \Phi_F(\mathcal{P}) = \Phi\}$. Given k probes, $|\mathcal{H}| \leq 4^k$ (each cell is in $\{0,1\}^2$). Each candidate $F \notin [N]_\mathcal{T}$ that is reachable by the autoformalizer has, by assumption, probability at least α_0 of being witnessed by each probe in $\mathcal{Q}$. Thus, with $k \geq (1/\alpha_0) \log(1/\delta\varepsilon)$ probes, the probability that any such F is consistent with $\hat{H}$ is at most $(1 - \alpha_0)^k \leq \delta\varepsilon$. Applying an Occam-style bound over the hypothesis class with size 4^k, and noting that $\log |\mathcal{H}| \leq 2k$ is dominated by the chosen k, gives the stated bound. □

A.4. Proof of Theorem 7.1

Let $\alpha = (\alpha(\Delta_\forall), \ldots, \alpha(\Delta_T))$ be the witnessability rates and w the class weights. Uniform allocation samples each class with probability $1/4$; APBA samples class D in proportion to $\alpha(D)w(D)$. The expected num-

ber of probes to witness any drift is $1/\bar{\alpha}_{\text{unif}}$ for uniform and $1/\bar{\alpha}_{\text{apba}}$ for APBA, where $\bar{\alpha}_{\text{unif}} = (1/4)\sum_D \alpha(D)$ and $\bar{\alpha}_{\text{apba}} = (\sum_D \alpha(D)w(D))^2/\sum_D \alpha(D)^2 w(D)^2$. The ratio is the heterogeneity factor γ, bounded below by $\max_D \alpha(D)/\min_D \alpha(D)$ when weights are equal. On DRIFTBENCH, $\max_D \alpha(D)/\min_D \alpha(D) \approx 2.0$, and with calibrated weights $\gamma \approx 3.2$. □

B. Structural Test for Convention Drift

Convention drift, by definition, leaves the consequence neighborhood invariant on standard probes. We address it through a structural test on the type signature of F, comparing it to a canonical-signature prediction derived from N. Given N, the canonical-signature predictor returns a tuple (Σ_N, K_N) where the first element Σ_N is the expected type signature (with universe levels and implicit arguments) and the second element K_N is a set of *structural keywords* (e.g. `Set`, `Finset`, `Function`, `Pi`) expected to appear. We then compare against the type signature of F: Compute the structural diff $\Delta\Sigma = \Sigma_F \triangle \Sigma_N$. Flag *convention* drift if $|\Delta\Sigma|$ exceeds a calibrated threshold but BPF returns FAITHFUL. In our experiments, this structural test catches an additional 0.04 F1 worth of convention-drift cases on the wild split. We view it as complementary to, not a substitute for, the main BPF procedure.

C. Case Studies

We summarize three case studies from the wild split; full Lean 4 code is in the supplementary materials. Case 1 (Banach–Steinhaus, $\Delta_\forall$): Natural-language statement: *"Let (T_i) be a family of bounded operators on a Banach space X such that for every $x \in X$, $\sup_i \|T_i x\| < \infty$. Then $\sup_i \|T_i\| < \infty$."* The candidate swapped the order of quantifiers in the conclusion. BPF generated a specialization probe instantiating $X = \ell^2$ and a specific bounded family for which pointwise boundedness holds, but the swapped conclusion fails; the entailment $F \to P$ is closed, flagging drift. The CPG probe targeting $\Delta_\forall$ recovered the witness with a 4-probe budget, where naive sampling was required 19. Case 2 (Continuity, Δ_H): Natural-language statement: *"A function continuous on a closed interval is uniformly continuous on that interval."* The candidate dropped the closedness hypothesis. A hypothesis-dropping probe instantiating $f(x) = 1/x$ on $(0, 1)$ the candidate refuted it. BPF flagged drift and FGD repaired the candidate on re-prompting. Case 3 (Convention drift, undetected): Natural-language statement: *"Every group homomorphism with a trivial kernel is injective."* The candidate formalized *the group* as `a monoid` (since group axioms beyond monoid structure are not used in the statement). BPF returns FAITHFUL because every probe is preserved; the structural test of Section B flags the `Monoid`/`Group` mismatch.

D. Probe-Generation Prompts

We include exact prompts used for the LLM-based probe generator. Each prompt is parameterized by the natural-language statement and constrained to emit a JSON-structured list of probe candidates with predicted labels. The full prompt suite, along with the system prompts and few-shot exemplars, is included in the supplementary materials.

E. Hyperparameters

Hyperparameter	Value
Probe budget k	32 (uniform) / 10 (APBA)
Tactic budget τ	30 s wall-clock
$\mathcal{E}$ accept threshold	0.93
$\mathcal{E}$ review threshold	0.78
Probe-class weights w	equal across $\{\Delta_\forall, \Delta_H, \Delta_C\}$, $1.5\times$ for Δ_T
Adversarial probes per statement	4
Specialization probes per statement	8
Hypothesis-drop probes per statement	up to 8
Boundary probes per statement	4
FGD candidates m	4 (default) / 8 (high-budget)