Munkres' General Topology Autoformalized in Isabelle/HOL

Dustin Bryant[1], Jonathan Julián Huerta y Munive[2], Cezary Kaliszyk[3], and Josef Urban[4]

[1] Independent, brydustin@gmail.com
[2] Aalborg University, jjhymuni@cs.aau.dk
[3] University of Melbourne, ckaliszyk@unimelb.edu.au
[4] AI4REASON and University of Gothenburg / Chalmers University
josef.urban@gmail.com

Abstract. We describe an experiment in LLM-assisted autoformalization that produced over 85,000 lines of Isabelle/HOL code covering all 39 sections of Munkres' *Topology* (general topology, Chapters 2–8), from topological spaces through dimension theory. The LLM-based coding agents (initially ChatGPT 5.2 and then Claude Opus 4.6) used 24 active days for that. The formalization is *complete*: all 806 formal results are fully proved with zero `sorry`'s. Proved results include the Tychonoff theorem, the Baire category theorem, the Nagata–Smirnov and Smirnov metrization theorems, the Stone–Čech compactification, Ascoli's theorem, the space-filling curve, and others.
The methodology is based on a "sorry-first" declarative proof workflow combined with bulk use of `sledgehammer`—two of Isabelle major strengths. This leads to relatively fast autoformalization progress. We analyze the resulting formalization in detail, analyze the human–LLM interaction patterns from the session log, and briefly compare with related autoformalization efforts in Megalodon, HOL Light, and Naproche. The results indicate that LLM-assisted formalization of standard undergraduate textbooks in Isabelle/HOL is quite feasible, cheap and fast, even if some human supervision is useful.

1 Introduction

Formalization of mathematics—the translation of informal textbook proofs into machine-checkable code—has typically required substantial manual effort. Major formal proof libraries such as Isabelle's Analysis library [19], Lean's Mathlib, and the Mizar Mathematical Library [2] represent years or decades of cumulative work by dedicated communities. The possibility of large language models significantly accelerating this process has thus attracted considerable interest.

In January 2026, the fourth author reported an experiment in which approximately 130,000 lines of formal topology were produced in roughly two weeks, formalizing large portions of Munkres' *Topology* [16] in the Megalodon proof system [22] using ChatGPT 5.2 via the Codex CLI. Despite being conducted in

a relatively lightweight system with limited prior training data, the experiment demonstrated that large-scale autoformalization is feasible in practice.

The present paper describes a related experiment:[5] the autoformalization of the same textbook in Isabelle/HOL [17], one of the field's most mature and well-supported proof assistants. In this setting, the LLM has access to Isabelle's extensive library (`Complex_Main`), powerful proof automation (`sledgehammer` [3], `blast`, `auto`, etc.), and the structured (declarative) Isar proof language. These features provide a substantially richer environment than in the Megalodon experiment. A natural question is whether such infrastructure significantly reduces the difficulty and speeds up the large-scale formalization. In practice, both effects are observed: the available automation and library support simplify and speed up some aspects of the development, while on the other hand, the more complex setting requires more involved oversight of the available tools.

Main results. Over 24 active days (March 2 – April 3, 2026), the project produced 85,472 lines of Isabelle/HOL across four chapter files, comprising 199 definitions and 806 lemmas, theorems, and corollaries covering all 39 sections (§§12–50) of point-set topology in Munkres' book. The formalization is **complete**: all results are fully proved with zero `sorry`'s. Chapter 1 (Set Theory and Logic) is covered by Isabelle's `Complex_Main`. Among the proved results are the Tychonoff theorem, the Baire category theorem (both parts), the Nagata–Smirnov and Smirnov metrization theorems, the Stone–Čech compactification, both the classical and general versions of Ascoli's theorem (Theorems 45.4 and 47.1—the latter depending on Tychonoff), Theorem 48.5 (pointwise limits on Baire spaces), the space-filling curve (Theorem 44.1), and the existence of a nowhere-differentiable function (Theorem 49.1).

Methodology. The experiment used two LLMs in sequence—ChatGPT 5.2 for bulk formalization and Claude Opus 4.6 for proof development—both driven by an automated tmux-based prompting loop. The central methodological idea is a "sorry-first" workflow: proof skeletons are written with placeholders, which are subsequently filled using automated reasoning tools such as `sledgehammer`. This separates proof *structure* from proof *search*.

Contributions. We produced: (1) a *complete* autoformalization of Munkres' general topology in a mainstream proof assistant—all 806 results fully proved with zero `sorry`'s. While we evolved the prompt and the methods, no part of the code has been written by a human (this is an intentional design choice of this experiment).[6] (2) A methodology for the long-running LLM–Isabelle interaction largely based on the sorry-first/sledgehammer workflow. (3) Analysis of the session log, revealing patterns of human–LLM interaction. (4) A comparison with several related autoformalization efforts on the same source material. (5) A qualitative evaluation of the mathematical adequacy of the formalization.

[5] The code is at `https://github.com/JUrban/isa_top_autoform1` .

[6] It is obviously possible to include human modifications and this will likely be done in various ways with this material in the future. However, direct and continued human modification usually makes the scientific results of autoformalization experiments and evaluations much less clear.

2 Setup

2.1 Source Material

The formalization target is a LaTeX representation of Munkres' *Topology* [16] (`top1.tex`, 7,956 lines), covering Chapters 2–8 (Sections 12–50): topological spaces and continuous functions (§§12–22), connectedness and compactness (§§23–29), countability and separation axioms (§§30–36), the Tychonoff theorem (§37–38), metrization and paracompactness (§§39–42), complete metric spaces and function spaces (§§43–46), and Baire spaces through dimension theory (§§47–50). The source contains 76 definitions, 109 theorems, 34 lemmas, 13 corollaries, and 115 examples; exercises are excluded from the formalization.

2.2 Isabelle/HOL

Isabelle [18] is a generic logical framework whose principal instantiation, Isabelle/HOL [17], provides higher-order logic with a large standard library. Our formalization only imports `Complex_Main`, giving it access to the reals and complex numbers, topological spaces (as a type class), metric spaces, continuity, compactness, Heine–Borel, measure theory, and extensive algebraic infrastructure. This is a considerable advantage over the Megalodon experiment, where the library consists of Brown's set-theoretic foundations with reals constructed via surreal numbers [6], but without the extensive analytical infrastructure that Isabelle provides.

The proof automation is correspondingly richer. Beyond `sledgehammer` [3]—which connects to external ATPs (Vampire, CVC5, Z3) with premise selection and proof reconstruction—Isabelle offers `blast` (classical tableaux), `auto`/`simp` (rewriting + classical reasoning), `linarith`/`presburger` (arithmetic decision procedures), and `meson`/`metis` (first-order provers). Megalodon has the `aby` hammer [6], which exports to TH0 TPTP [21] and has good ATP performance but lacks premise selection and proof reconstruction—a critical limitation for the LLM workflow.

2.3 LLM Agents

The project used two LLMs in sequence:
Phase 1 (March 2–13): ChatGPT 5.2 with high reasoning, via OpenAI's Codex CLI—the same model and interface as the Megalodon experiment. This phase (running continuously on a server) produced 268 commits ("topology 0001" through "topology 0282"), formalizing all definitions and theorem statements with proofs often deferred via `sorry`.
Phase 2 (March 21 – April 3): Claude Opus 4.6 via Claude Code (Anthropic's CLI agent—running with pauses on the fourth author's notebook), focused on systematic proof development. This phase produced 884 commits with detailed theorem-by-theorem progress, reducing the sorry count from 51 to **zero**.

Both agents ran in an automated tmux-based loop: a script monitors the session and re-issues the prompt ("Read CLAUDE.md and follow instructions")

whenever the agent finishes or stalls—the same mechanism used in the Megalodon experiment. The human occasionally intervened for focused direction or to correct misbehavior (see Section 5).

2.4 Tooling Evolution

As the project progressed, several infrastructure improvements were developed. A session-splitting optimization—caching Chapters 2–4 in a separate precompiled heap image—cut incremental build times from 50–60 s to 12–13 s. Since our workflow is compilation based, this is critical for fast feedback and progress. Performance profiling revealed that slow `metis` calls in equality-free topology proofs could be replaced by `meson`, often yielding 10× speedups. Custom Isabelle CLI tools were developed for the project: `eval_at` (with per-line timing via the `-t` flag) and `desorry` (for automated sorry elimination)[7]. Together with Isabelle's built-in `process_theories` command (which runs theory processing outside the session timeout), these tools enabled the bulk sledgehammer workflow described in the next section.

3 The Sorry-First Methodology

The central methodological insight, hard-won through repeated encounters with Isabelle's capacity for combinatorial explosion, is the *sorry-first workflow* (see also Fig 1):

1. Write the proof skeleton with `sorry` at *every* step. No other tactic is permitted in new code—"not `by fast`, not `by linarith`, not `by simp`, not `by (rule ...)`, not even `done`," as the final version of the instructions puts it, reflecting successive refinements of the instruction set.
2. Build to verify the skeleton is well-formed.
3. Annotate all sorry'd steps with `sledgehammer [timeout = 10]`.
4. Run `process_theories` *once* to collect all ATP suggestions in a single pass.
5. Substitute the fastest proof for each step.
6. When sledgehammer fails, decompose the step into finer `have` blocks and repeat.
7. Use `eval_at -t` to profile and optimize slow tactics.

This rule is the culmination of a rather painful evolutionary process. The models have a natural tendency to write proof methods immediately.[8] Unfortunately, in a context importing `Complex_Main`—which contributes thousands of simplification rules—even an innocent-looking `by auto` can trigger minutes of

[7] Both available at `https://github.com/yonoteam/isa_agentic_cli_tools`

[8] This seems to be an artifact of the unconstrained LLM training on the large Isabelle corpora available online. It is interesting to note – as a counterexample to the accepted LLM "token prediction is all you need" wisdom – that an (L)LM trained on a modified corpus with the proof methods removed would be very useful and likely much better behaved for our approach.

Table 1. Formalization content by chapter (199 definitions and 806 results overall).

File	Defs	Results	File	Defs	Results
Ch2 (§§12–22)	78	189	Ch4 (§§30–36)	18	123
Ch3 (§§23–29)	28	107	Ch5–8 (§§37–50)	75	387

fruitless search. The instruction file (`CLAUDE.md`) went through eight revisions, each prompted by a specific failure mode (see Section 5).

One consequence of this workflow is that the LLM did not extend the simplifier set in any way. Similarly, it did not add any other automation-related attributes to any theorems (such as `[intro]` or `[elim]`). All 8,160 `unfolding` calls and the dominance of explicit `by blast` (8,879 uses) over `by auto` (576 uses) reflect this: each definition is manually unfolded at every use site, and it is not registered for automatic simplification. This is *safe* as it prevents cascading simpset explosions but it is also non-standard Isabelle practice, where definitions enable working with derived properties at a higher level of abstraction.

4 The (Auto-)Formalization

4.1 Design

Unlike most previous formalizations of topology that rely on type classes [9], type definitions [19], or locales [5], the LLM chose (independently of the humans) to use *explicit carrier sets*, e.g. a topology as a pair $(X, \mathcal{T})$ with $\mathcal{T} \subseteq \mathcal{P}(X)$ is formalized as shown below.

```
definition is_topology_on :: 'a set ⇒ 'a set set ⇒ bool where
"is_topology_on X T ⟷
{} ∈ T ∧ X ∈ T ∧ (∀U. U ⊆ T ⟶ ⋃U ∈ T) ∧
(∀F. finite F ∧ F ≠ {} ∧ F ⊆ T ⟶ ⋂F ∈ T)"
```

This mirrors Munkres' definitions and is closer to more recent topology formalizations [5] that do not interoperate with Isabelle's type-class-based topology (which allows only one topology per type). Names of lemmas and definitions in the formalization include a `top1_` prefix (again, a decision made by the LLM). They follow Munkres' pattern `Theorem_N_M` and each is annotated with its source location as follows:

```
(** from §37 Theorem 37.3 (Tychonoff) [top1.tex:5253] **).
```

4.2 Content

Table 1 summarizes the formalization. The 806 formal results include 195 directly corresponding to Munkres' numbered results (covering all 156 non-exercise statements in the source, with some split into parts) and 611 helper lemmas (a helper-to-Munkres ratio of ≈3.1:1).

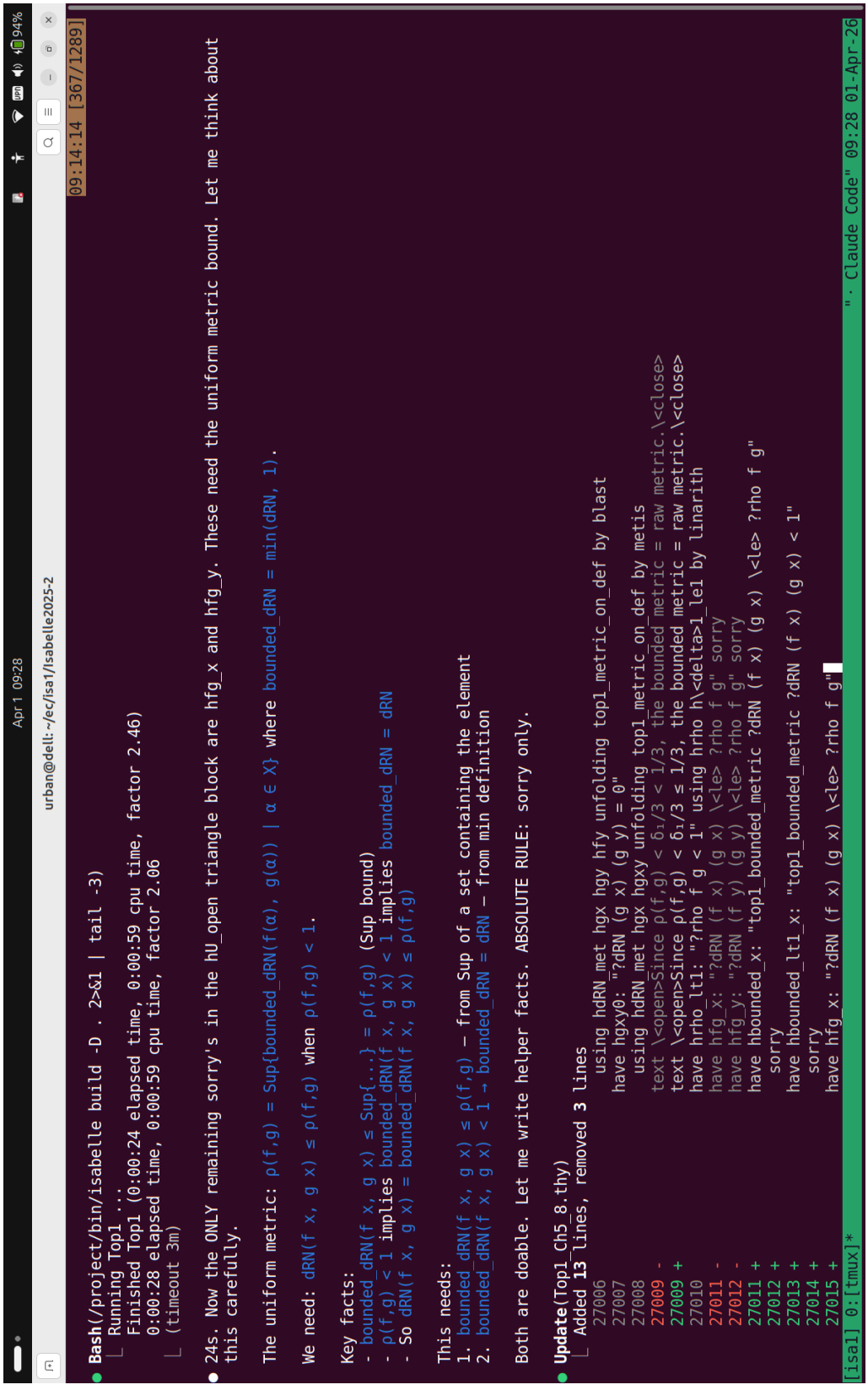

Fig. 1. Example of the LLM's strategic thinking and use of the sorry-based method.

The initial Chapters 2–4 cover among other things: the Urysohn lemma (Theorem 33.1), the Urysohn metrization theorem (Theorem 34.1), the Tietze extension theorem (Theorem 35.1), and the imbedding of manifolds (Theorem 36.2). A comparison of such landmark proofs between different projects and systems is instructive, but one must be careful about what is being counted. In our formalization, the proofs are *heavily factored* into helper lemmas, so there are two natural measurements: the *direct* proof body of the theorem itself, and the entire *section* including all supporting infrastructure. The Megalodon paper [22] reports *direct* proof lengths (the theorem's own proof body, Table 3 therein); Mulligan and Paulson [14] report a self-contained 275-line proof.

Table 2 gives both measurements for our formalization, alongside the Megalodon and Mulligan–Paulson figures. The direct proof comparisons are as follows: our Urysohn lemma (Theorem 33.1) is 287 lines versus Megalodon's 2,964; our Tietze extension (Theorem 35.1) is 324 lines versus Megalodon's 10,369—the infamous "Battle of the Tietze Hill," which required ∼20 hours and 35 iterations in Megalodon [22]. These order-of-magnitude differences in direct proof length reflect two factors: (i) Isabelle's `Complex_Main` supplies analytical prerequisites that Megalodon must build inline, and (ii) our factoring strategy moves a lot of content into helper lemmas (the section totals of 3,126 and 3,597 lines are closer to the Megalodon figures). In effect, the same mathematical content is present but distributed differently—concentrated in one large proof in Megalodon, versus spread across 18–20 small lemmas in Isabelle.

The Mulligan–Paulson proof of the Urysohn metrization theorem (275 lines) is much shorter than both our direct proof (1,196 lines—a single structured proof block with 260 `have` steps that was not factored into helpers) and Megalodon's (2,174 lines). This reflects their use of the type-class library where `metrizable_space`, `regular_space`, and `second_countable` are pre-existing concepts with substantial lemma infrastructure—our formalization defines these from scratch with explicit carrier sets. Our Theorem 34.1 is also a case where the sorry-first factoring discipline was not consistently applied: the monolithic proof was produced in Phase 1 by ChatGPT 5.2, before the sorry-first methodology was adopted.

The contrast between "Direct" and "Section" columns reveals the factoring strategy: the Baire category theorem's direct proof is just 62 lines, but its section is 2,907 lines because the heavy lifting is in the nested-ball construction helper (`top1_baire_nested_ball_helper`) and the compact-Hausdorff case. Similarly, Theorem 49.1's direct proof is 53 lines—it merely assembles results from the 24 supporting lemmas that do the real work. This extreme decomposition is a direct consequence of the sorry-first methodology: each helper is independently proved via sledgehammer, and the main theorem becomes a short assembly of pre-verified pieces.

Table 3 gives a broader view, listing all theorems with a direct proof of over 300 lines across the entire formalization. The longest direct proof is the Urysohn metrization theorem (Theorem 34.1, 1,196 lines)—a monolithic Phase 1 proof that predates the factoring discipline. The second longest is the dimension-theory

Table 2. Proof sizes for landmark theorems

Theorem	Direct	Section	Helpers	Megal.	M–P
Urysohn lemma (33.1)	287	3,126	20	2,964	—
Urysohn metriz. (34.1)	1,196	3,373	19	2,174	275
Tietze extension (35.1)	324	3,597	18	10,369	—
Tychonoff (37.3)	198	1,394	6	—	—
Nagata–Smirnov (40.3)	968	2,455	38	—	—
Smirnov metriz. (42.1)	83	347	4	—	—
Space-filling curve (44.1)	325	1,919	44	—	—
Ascoli general (47.1)	372	1,912	29	—	—
Baire category (48.2)	62	2,907	21	—	—
Thm 49.1 (nowhere-diff.)	53	1,296	24	—	—
Dim. imbedding (50.5)	1,103	6,271	96	—	—

Direct = theorem's own proof body (comparable to Megalodon column). Section = full section including all helper lemmas and definitions. Megal. = Megalodon direct proof length (from [22], Table 3). M–P = Mulligan–Paulson (self-contained, type-class-based).

imbedding theorem (Theorem 50.5, 1,103 lines), involving general-position arguments and partitions of unity. Among other Phase 2 proofs, the Nagata–Smirnov theorem (968 lines) and Theorem 48.5 (833 lines) are the largest. The median direct proof length for the 806 results is approximately 25 lines, reflecting the prevalence of short, focused lemmas with just 2–5 `have` steps each.

4.3 Completeness

As of April 3, 2026, the formalization contains **zero** `sorry`'s: all 806 formal results across all 39 sections are fully proved. The last sorry was eliminated in Corollary 45.5 (characterizing compact subsets of complete metric spaces as closed and bounded). The final push (March 30 – April 3) required 355 commits to resolve the remaining gaps, which included the space-filling curve (§44), the sup-uniform topology equality (§46), Ascoli's theorem (§47), and the dimension-theory imbedding theorems (§50). The space-filling curve construction, involving a Hilbert-curve-style recursive definition with snake-order cell traversal, was particularly challenging and required 44 commits on April 3 alone.

4.4 Major Proved Theorems

Table 4 lists the major proved theorems across *all* chapters. Chapters 2–4 were proved during Phase 1 (by ChatGPT 5.2); Chapters 5–8 during Phase 2 (by Claude Opus 4.6).

The Nagata–Smirnov metrization theorem (Theorem 40.3) deserves special mention as one of the most complex proofs in the formalization: 2,455 lines, 38 helper lemmas, requiring uniform convergence arguments, Urysohn-style constructions for G_δ sets, and a metric defined as a supremum of weighted function

Table 3. All direct proofs exceeding 300 lines

§	Theorem	Lines	§	Theorem	Lines
34	Thm 34.1 (Urysohn met.)	1,196	38	Thm 38.2 (Stone–Čech)	335
50	Thm 50.5 (dim. imbedding)	1,103	44	Thm 44.1 (space-filling)	325
21	Lem. 21.4 (seq. closure)	990	33	Thm 33.2 (product)	325
40	Thm 40.3 (Nagata–Sm.)	968	29	Thm 29.2 (1-pt compact.)	325
23	Thm 23.6 (fin. prod. conn.)	900	35	Thm 35.1 (Tietze)	324
32	Thm 32.4	839	45	Thm 45.4 (Ascoli classical)	319
48	Thm 48.5 (ptw. limits)	833	25	Thm 25.2 (components)	318
32	Thm 32.1	652	32	Thm 32.3	313
46	Thm 46.7	577	19	Thm 19.5 (product)	311
46	Thm 46.11	562	40	Lem. 40.1 step 3	308
39	Lem. 39.2	561	13	Lem. 13.4 ($\mathbb{R}_\ell, \mathbb{R}_K$)	307
30	Thm 30.2 (1st cnt. prod.)	500	19	Thm 19.5 (box)	304
37	Lem. 37.2 (max. FIP)	498	22	Thm 22.2 (quotient)	297
30	Thm 30.2 (2nd cnt. prod.)	473	19	Thm 19.2 (product)	294
32	Thm 32.2	458	26	Thm 26.7	294
17	Thm 17.11	436	33	Thm 33.1 (Urysohn)	287
22	Thm 22.1 (quotient)	427	34	Thm 34.2 (imbedding)	270
18	Thm 18.1 (continuity)	426	31	Thm 31.2	270
36	Thm 36.1 (manifold)	406	43	Thm 43.5 (complete Y^J)	257
47	Thm 47.1 (Ascoli general)	372	43	Lem. 43.3	256
38	Thm 38.4	371	40	Lem. 40.1 step 1	249
15	Thm 15.2 (product basis)	369	18	Thm 18.2 (pasting)	326
26	Thm 26.3 (closed compact)	362	40	Lem. 40.2 (Urysohn G_δ)	342
41	Thm 41.7 (partition unity)	347	38	Thm 38.5	232

differences. Its development spanned 22 hours across two days. The space-filling curve (Theorem 44.1) and the dimension-theory imbedding (Theorem 50.5) are comparably involved, each requiring substantial novel infrastructure—recursive curve constructions and general-position arguments, respectively—that goes well beyond routine topology.

We show the formal statements of these three results, first in mathematical shorthand and then as they appear in the Isabelle sources.

Nagata–Smirnov metrization (Theorem 40.3). Metrizable iff regular with a σ-locally-finite basis:

```
theorem Theorem_40_3:
shows "top1_metrizable_on X TX ⟷
(is_topology_on X TX ∧ top1_regular_on X TX ∧
(∃B. top1_sigma_locally_finite_family_on X TX B ∧ basis_for X B TX))"
```

Space-filling curve (Theorem 44.1). There exists a continuous surjection $[0,1] \to [0,1]^2$:

```
theorem Theorem_44_1:
shows "∃f::real ⇒ (real × real).
```

Table 4. Selection of major fully proved theorems (all chapters)

§	Theorem	Description	Ch
13	Lem. 13.1–13.4	Bases and topology generation	2
16	Thm 16.3–16.4	Subspace topology	2
17	Thm 17.1–17.11	Closure, interior, Hausdorff, T1	2
18	Thm 18.1–18.4	Continuous functions (all characterizations)	2
19	Thm 19.1–19.6	Product topology (box & product)	2
20	Thm 20.1–20.5	Metric topology, uniform metric	2
22	Thm 22.1–22.2	Quotient topology	2
23	Thm 23.3–23.6	Connected spaces	3
24	Thm 24.1, 24.3	Connected subspaces of $\mathbb{R}$, IVT	3
26	Thm 26.2–26.9	Compact spaces, FIP, tube lemma	3
27	Thm 27.1–27.7	Heine–Borel, Lebesgue number, uniform cont.	3
29	Thm 29.1–29.2	Local compactness, 1-pt compactification	3
32	Thm 32.1–32.4	Normal spaces	4
33	Thm 33.1–33.2	Urysohn lemma, completely regular	4
34	Thm 34.1–34.3	Urysohn metrization theorem	4
35	Thm 35.1	Tietze extension theorem	4
36	Thm 36.1–36.2	Imbedding of manifolds	4
37	Thm 37.3	Tychonoff theorem	5
38	Thm 38.2–38.5	Stone–Čech compactification	5
40	Thm 40.3	Nagata–Smirnov metrization	6
41	Thm 41.1–41.8	Paracompactness (all results)	6
42	Thm 42.1	Smirnov metrization	6
43	Thm 43.2–43.7	Complete metric spaces (all results)	7
45	Thm 45.1, 45.4	Compactness in metric spaces	7
46	Thm 46.1–46.11	Pointwise & compact convergence	7
44	Thm 44.1	Space-filling curve	7
45	Thm 45.1, 45.4, Cor. 45.5	Compact $\Leftrightarrow$ complete+TB; Ascoli (classical)	7
47	Thm 47.1	Ascoli's theorem (general, uses Tychonoff)	7
48	Thm 48.2, 48.5	Baire category, pointwise limits	8
49	Thm 49.1	Nowhere-differentiable function	8
50	Thm 50.1–50.6	Dimension theory, imbedding	8

```
top1_continuous_map_on
(top1_closed_interval 0 1)
(top1_closed_interval_topology 0 1)
((top1_closed_interval 0 1) × (top1_closed_interval 0 1))
(product_topology_on
(top1_closed_interval_topology 0 1)
(top1_closed_interval_topology 0 1)) f
∧ f ‘ (top1_closed_interval 0 1)
= (top1_closed_interval 0 1) × (top1_closed_interval 0 1)’’
```

Ascoli's theorem, general version (Theorem 47.1). Given a topological space $(X, \mathcal{T}_X)$, a metric space (Y, d), and $\mathcal{F} \subseteq \mathcal{C}(X, Y)$: the forward direction

states that equicontinuity of $\mathcal{F}$ plus pointwise compact closures implies compact closure in the compact-convergence topology; the converse holds when X is locally compact and Hausdorff. The full Isabelle statement, with its deeply nested subspace topologies, is:

```
theorem Theorem_47_1:
assumes "is_topology_on X TX"
assumes "top1_metric_on Y d"
assumes "ℱ ⊆ top1_continuous_funcs_on X TX Y
(top1_metric_topology_on Y d)"
shows "(top1_equicontinuous_family_on X TX Y d ℱ
∧ (∀a∈X. top1_compact_on
(closure_on Y (top1_metric_topology_on Y d)
((λf. f a) ` ℱ))
(subspace_topology Y (top1_metric_topology_on Y d)
(closure_on Y (top1_metric_topology_on Y d)
((λf. f a) ` ℱ)))))
⟶ (∃K. ℱ ⊆ K ∧ top1_compact_on K
(subspace_topology
(top1_continuous_funcs_on X TX Y (top1_metric_topology_on Y d))
(subspace_topology (top1_PiE X (λ_. Y))
(top1_compact_convergence_topology_on X TX Y d)
(top1_continuous_funcs_on X TX Y (top1_metric_topology_on Y d)))
K))"
and "(top1_locally_compact_on X TX ∧ is_hausdorff_on X TX)
⟶ (∀K. top1_compact_on K (...)
⟶ top1_equicontinuous_family_on X TX Y d K
∧ (∀a∈X. top1_compact_on
(closure_on Y ... ((λf. f a) ` K)) ...))"
```

The backward direction (second `and` conjunct) mirrors the forward direction with K in place of $\mathcal{F}$; its subspace topology nesting is identical and elided here for space.

5 Session Log Analysis

The Claude Code session log (1.3 GB compressed, 92,524 JSONL records) provides a detailed record of the Phase 2 autoformalization. We used LLM-developed analysis tools (`analyze_session.py`) to extract quantitative data from the log.

5.1 Scale

The session comprises 50,119 assistant messages containing 17,925 Bash commands (of which 5,186 are `isabelle build` and 3,266 are `process_theories` runs), 7,135 file edits, and 4,962 file reads. The `ccusage` tool reports 2.66 M output tokens and 17.2 billion cache-read tokens, corresponding to an estimated API cost of $8,895 at list prices for Claude Opus 4.6. The actual experiment ran on a Pro/Max subscription ($200/month), making the effective cost approximately $160. This highlights the scale and cost-efficiency of the approach.

5.2 Automation and Human Intervention

Of 764 non-system user messages, 635 (83.1%) were automatically issued prompts "Read CLAUDE.md". The remaining 129 were manual interventions, 59 of them were corrections—the human telling the LLM to stop undesirable behavior. The correction themes, in decreasing frequency:

- *Tactic explosion* (9): the LLM inserting uncontrolled `blast` or `auto` calls, causing timeouts. Representative: "how did you again manage to smuggle in uncontrolled slow `by` calls??? we were through this multiple times."
- *Inefficient tool use* (6): running Isabelle repeatedly instead of capturing output once.
- *Cherry-picking easy goals* (4): "please stop jumping around looking for low hanging fruit; we have to do it all."
- *Resource management* (3): backgrounding Isabelle processes that consume memory.[9]

These corrections drove the CLAUDE.md evolution: recurring failures were turned into rules in the instruction file. In particular, the "ABSOLUTE RULE" (sorry-first workflow) was added on March 24 after repeated tactic-explosion incidents.

5.3 Session Duration

The 626 automated sessions had a median duration of 13.0 minutes (mean 24.5, max 263). The distribution is right-skewed: most sessions last under 20 minutes, but some run for over 90 minutes autonomously. The longest session (263 minutes, over 4 hours) suggests that with appropriate instructions, the LLM can sustain extended autonomous work sessions.

6 Comparison with Related Efforts

Since the first Megalodon experiment, we and others have used the same Munkres material for comparing the methods and systems.

6.1 The Megalodon Experiment

We estimate that the Isabelle formalization is roughly half the length, primarily because the library of Isabelle/HOL provides infrastructure that Megalodon must build from scratch. The number of remaining gaps is substantially smaller, largely thanks to `sledgehammer`'s premise selection and proof reconstruction. Both experiments use the same automation loop and the same textbook, making

[9] This is dangerous because runaway Isabelle jobs can silently keep allocating RAM, and multiple background processes may write to the same heap images concurrently, corrupting or destroying the cached heaps and forcing expensive rebuilds.

Table 5. Comparison with the Megalodon experiment [22]

	Isabelle/HOL	**Megalodon**
Logic	HOL	Higher-order set theory
LLMs	ChatGPT 5.2 + Claude 4.6	ChatGPT 5.2
Library	Complex_Main (extensive)	Set theory + reals
Automation	sledgehammer + blast/auto	aby (THF, no reconstruction)
Active days	24	∼14
Lines	85,472	∼130,000
Remaining gaps	**0**	significant
Sections	§§12–50 (39)	§§12–50 (39)

this perhaps the cleanest cross-system comparison of LLM-assisted formalization to date.[10]

Additionally, in Megalodon, the LLM must construct detailed declarative proofs largely on its own, as the `aby` hammer lacks proof reconstruction. In Isabelle, the sorry-first workflow means the LLM focuses on *proof structure* while the ATPs handle *proof steps*. This division of labor—proof structure from the neural network, and proof completion by the ATPs/hammers—appears to be more effective than asking the LLM to do both.

6.2 Other Concurrent Work on Munkres and Beyond

Harrison (with initial setup by Lee and the fourth author) has been carrying out autoformalizations in HOL Light using a similar approach.[11] This spans not just topology but e.g. an entire textbook [24] on probability theory.[12]

Mulligan and Paulson [14] used Amazon's Isabelle/Assistant [15] that integrates LLMs into Isabelle/jEdit. Paulson used Claude Opus 4.6 to prove the Urysohn metrization theorem in approximately 2.5 hours, following Munkres' proof—the agent eventually finishing autonomously. Their 275-line proof (using Isabelle's type-class library) provides an instructive contrast with explicit-carrier-set approaches [5] like ours.

Hahn and De Lon [8] have submitted an autoformalization of the Urysohn lemma in Naproche/Felix to ITP 2026, showcasing the approach in a controlled-natural-language proof checker.

6.3 What Each System Does Better

The Megalodon experiment demonstrates that LLM-assisted formalization can work even with minimal library and automation support. Megalodon's simplicity

[10] Note however that the Megalodon formalization also includes exercises, which we largely managed to avoid here. A detailed comparison may be done when the Isabelle formalization is completed.

[11] `https://github.com/jrh13/hol-light/commit/c845dd3bcc09a8`

[12] `https://github.com/jrh13/hol-light/commit/1c7690ec12319e`

(just one fast compilation-style checker) and its very transparent (even if verbose) proofs prevent the LLMs from getting lost in complicated tool usage and proofs where the advanced tactics may blow up.

The Isabelle experiment benefits from stronger automation (sledgehammer) and a richer library, resulting in shorter code and fewer remaining gaps. The price is that the more complicated setting can go wrong more often and required so far more supervision. The Mulligan–Paulson approach (human-assisted, IDE-integrated, type-class-based) produces more idiomatic Isabelle code that interoperates with the existing library. Harrison's HOL Light work leverages that system's high flexibility combined with simpler language (no type classes etc) than Isabelle/HOL. It seems that no single approach dominates but this is an interesting and evolving research area.

7 Proof Engineering Observations

7.1 Tactic Selection

The formalization contains 8,879 uses of `by blast`, 4,608 of `by simp`, and only 576 of `by auto`—reflecting the CLAUDE.md prohibition on unrestricted automation. The 8,160 `unfolding` calls result from the LLM never registering simp rules. There are 100 uses of `by metis` versus 59 of `by meson`; we observed that in general topology (where most reasoning is equality-free), `meson` is typically 10× faster than `metis`, but the LLM lacks the intuition for when to use which.

7.2 Proof Structure

The proofs are remarkably uniform in style: over 16,000 `have` steps, 4,702 `show` steps, about 2,000 `proof` blocks, and 1,960 `obtain` steps. Each intermediate result is explicitly named (`have hfoo: "..."`) and supplied to the closing tactic via `using`. This is verbose but debuggable—a trade-off the CLAUDE.md rules explicitly mandate.

The helper-to-Munkres ratio of 3.1:1 means each textbook theorem spawns, on average, three additional lemmas. The Nagata–Smirnov theorem, at 38 helpers, is the extreme case; the Tychonoff theorem, with 6, is more typical.

7.3 Build Time as a Design Constraint

The 120-second session timeout (demanded by the CLAUDE.md rules) acts as a hard constraint on proof complexity. Several commits document performance crises: "Ch5_8 from 115s to 11.5s cumulated" records a 10× speedup achieved by replacing slow `blast` calls with `meson` alternatives found by `sledgehammer`. The session split (caching Chapters 2–4) provided a further 4–5× improvement. We note that the need for "performance engineering" in theorem proving may be a bit unusual concern, but it becomes unavoidable in our compilation-based setting when the LLM generates proofs at scale and without the human interaction.

7.4 The Tychonoff Theorem: A Case Study

The Tychonoff theorem (Theorem 37.3: arbitrary products of compact spaces are compact) was the first major result proved in Phase 2 (March 21, 07:52). The proof follows Munkres' approach via the finite intersection property (FIP) and Zorn's lemma, decomposed into six helper lemmas:

```
theorem Theorem_37_3:
assumes hIne: "I ≠ {}"
assumes hComp: "∀i∈I. top1_compact_on (X i) (TX i)"
shows "top1_compact_on (top1_PiE I X)
(top1_product_topology_on I X TX)"
```

Key helpers include `Lemma_37_1` (existence of maximal FIP collections via Zorn's lemma), `Lemma_37_2` (properties of maximal FIP collections), as well as the lemma `tychonoff_subbasis_in_maxFIP` (subbasis elements belong to the maximal FIP extension). The proof involves delicate combinatorics of product topologies, subbases, and Hilbert's ε operator (`SOME`)—the latter being a recurring source of difficulty in Isabelle formalizations, as the chosen witness must be shown to satisfy the required properties across multiple contexts.

7.5 The Nowhere-Differentiable Function: From Baire to Analysis

Theorem 49.1—the existence of a continuous function on $[0, 1]$ that is nowhere differentiable—was one of the most technically demanding results, fully completed on March 25 in a 14-hour, 40-commit session. The proof applies the Baire category theorem to the complete metric space $\mathcal{C}([0, 1])$ with the sup metric. The key steps: (i) prove $\mathcal{C}([0, 1])$ is complete (`C01_complete`), hence Baire; (ii) define the difference-quotient sets U_n and prove they are open (`top1_U49_open`) and dense (`U49_dense_approx`, requiring piecewise-linear approximation via "triangle functions"); (iii) assemble via the Baire property. The density proof required uniform continuity (Heine–Cantor), Archimedean arguments for the mesh size, and careful partition-of-unity reasoning for the piecewise approximant—a substantial piece of formal analysis sitting atop the topology.

8 Qualitative Analysis of the Definitions and Theorems

Beyond the quantitative metrics reported above, we conducted a manual review of the formalization's definitions, theorem statements, and proofs, checking faithfulness to Munkres, mathematical correctness, and adherence to Isabelle best practices. The review was based on commit `06dd7b2` (March 31), since the human reviewers needed a fixed snapshot to work from while the formalization continued to evolve. The review covered Chapters 2–4 in detail (approximately 53,000 lines) and sampled the remaining chapters. We summarize the findings below, organized by theme.

8.1 Definitional Soundness

The most consequential class of issues concerns definitions that are *logically weaker* than the intended mathematical concept, typically because a carrier-set constraint is missing from the right-hand side.

The topology predicate itself. The formalization defines `is_topology_on X T` without requiring $T \subseteq \mathcal{P}(X)$. This omission is not merely cosmetic: one can formally derive, for instance, that the collection $T := \{\emptyset, \{0\}, \{1\}, \{0,1\}\}$ satisfies `is_topology_on {0} T`, despite containing $\{1\} \not\subseteq \{0\}$. The root cause is Isabelle's convention that $\bigcap \emptyset = \mathit{UNIV}$, which forces the clause $F \neq \{\}$ in the finite-intersection axiom; this in turn means the definition does not prevent T from containing sets that extend beyond X. A corrected definition would add the premise $T \subseteq \mathit{Pow}\ X$, restoring the standard requirement that every open set is a subset of the carrier.

Propagation to derived predicates. The same weakness propagates to several downstream definitions. The predicate `openin_on X T U` is defined as $U \in T$ without requiring $U \subseteq X$, allowing formally "$\{1\}$ is open in the space $\{0\}$." Similarly, `neighborhood_of` does not constrain its argument to lie within X, and the finer-than relation `finer_than T T'` neglects to require that both T and T' are topologies on the same underlying set—making it meaningful to compare arbitrary set families. The product-basis and box-basis definitions carry redundant conditions (e.g., requiring both $U_i \in T_i$ and $U_i \subseteq X_i$, when the latter follows from the former under the assumption that each T_i is a topology on X_i); while not incorrect, this bloat obscures the mathematical content.

Assessment. Crucially, these definitional weaknesses do *not* invalidate the proved theorems: every theorem in the formalization carries explicit `is_topology_on` assumptions that supply the missing constraints at each use site. The proofs are therefore sound, but the definitions are less reusable than they could be—a user inheriting these definitions without the accompanying assumptions could derive spurious results. This pattern is characteristic of LLM-generated code: the model reliably produces definitions that are *sufficient* for the proofs at hand but does not anticipate downstream reuse or defensive design.

8.2 Faithfulness to Munkres

A second category of issues concerns theorem statements that deviate from the textbook in content, not merely in formulation.

Incomplete statements. Several theorems omit mathematically significant clauses present in Munkres. For example, Theorem 19.1 in the formalization lacks the characterization that in the product topology, all but finitely many factors equal the full space X_α—the defining feature that distinguishes the product topology from the box topology. Theorem 26.7 proves compactness of a binary product $X \times Y$ but does not state the finite-product generalization that Munkres gives (and which follows by a straightforward induction). Theorem 15.1 is reduced to the assertion that the product topology is a topology, omitting the stronger statement that products of basis elements form a basis for it.

Mislabeled results. In a few cases, a lemma is labeled with a Munkres number but proves a different (often weaker or tangentially related) statement. Most notably, `Lemma_23_1` in the formalization merely unfolds the definition of connectedness, whereas Munkres' Lemma 23.1 characterizes connectedness of a subspace Y in terms of sets whose closures do not meet the complementary part of Y—a genuinely different result. Similarly, Lemma 23.2 omits the disjointness clause ($Y \cap D = \emptyset$ or $Y \cap C = \emptyset$) that gives the result its geometric content.

Missing content. The formalization omits several standard topics from this part of Munkres: the alternate characterization of connectedness via clopen sets, sequential compactness and Theorem 28.2, the section on nets, and the notion of "distance from a point to a set." It also omits several counterexamples showing that limit-point compactness does not imply compactness. These gaps are understandable given the project's pace, but they affect the formalization's value as a standalone reference.

8.3 Proof Quality

The proofs exhibit a characteristic uniformity imposed by the sorry-first methodology. While this discipline is effective at scale, it introduces several systematic shortcomings.

Missed simplifications. Many lemmas that serve as "wrappers" around previously proved results retain unnecessarily long proofs when a one-line combination would suffice. For example, `Lemma_21_2`—which simply conjoins its two constituent halves `Lemma_21_2_sequence` and `Lemma_21_2_sequence_converse`—could be discharged by two `metis` calls rather than a structured Isar proof. Several proofs throughout can be replaced by short `by (simp add: ...)` or `by (smt ...)` invocations. The sorry-first workflow, which delegates every step to `sledgehammer` in isolation, does not attempt global proof compression after the fact.

Tactic-level inefficiencies. Isolated instances of the deprecated `apply` / `using` idiom persist (e.g., `apply (rule bexI[...])` followed by `using ... by blast`), where a single combined `by (rule ...)` call would be both shorter and more idiomatic. Duplicate rewrite-rule warnings (`simp add:` supplying a rule already in the simpset) appear throughout, suggesting that `sledgehammer`'s suggestions were accepted without filtering. These are minor blemishes individually but accumulate across 85,000 lines.

No proof optimization pass. Because the methodology prioritizes *completing* all proofs over *polishing* them, the formalization lacks the compression pass that a human formalizer would perform: merging small helper lemmas back into their parent proofs, shortening verbose Isar scripts, and eliminating redundant intermediate steps. A number of long proofs (e.g., Theorem 30.3) could be substantially shortened with manual attention.

8.4 Structural and Organizational Issues

Ordering. Munkres introduces separations before defining connectedness; the formalization reverses this order. Several lemmas appear far from their natural location—for example, the basis-monotonicity lemma `topology_generated_by_basis_mono_via_basis_elems` appears in §21 rather than §13 where bases are introduced. These placement issues are mostly cosmetic but make the formalization harder to navigate.

Redundant definitions. The predicate `countable` is defined three times: once locally in Chapter 3, once as `top1_countable` in Chapter 4, and it is also available as the identical definition in `HOL-Library.Countable_Set`. A bridging lemma `top1_countable_iff_countable` confirms the equivalence, underscoring the redundancy.

Unnecessary assumptions. A recurring pattern is the inclusion of assumptions that are logically redundant given the other hypotheses. For example, the assumption `is_topology_on` is redundant in lemmas about path reversal where the continuity hypothesis already implies it, and the assumption $a \leq b$ in Theorem 27.1 when the conclusion is vacuously true for $b < a$. While harmless, these weaken the stated results and again reflect the LLM's tendency to include "safe" extra hypotheses rather than proving the sharpest possible statement.

8.5 Library Integration

The formalization's relationship with Isabelle's existing topology library is complex. On one hand, importing `Complex_Main` provides powerful analytical infrastructure; on the other, the explicit-carrier-set design deliberately avoids the type-class-based topology, creating a parallel universe of definitions.

Several sections—notably §24 (connected subspaces of $\mathbb{R}$), §27 (compact subspaces of $\mathbb{R}$), and the path-related definitions—awkwardly mix the two systems, using Isabelle's built-in `topological_space` type class for reasoning about $\mathbb{R}$ while maintaining explicit carrier sets for the general theory. In at least one case (`top1_compact_Icc_linear_continuum`), a lemma is stated entirely in the language of Isabelle's existing type-class topology with no reference to the project's own definitions.

A reconciliation layer—systematically relating `is_topology_on` to Isabelle's `topological_space` class—would be valuable but was not attempted.

8.6 What Went Well

It would be misleading to focus solely on deficiencies. Several aspects of the formalization are genuinely impressive, especially given the 24-day timeline.

Coverage and correctness. All 156 non-exercise numbered results from Munkres' Chapters 2–8 have formal counterparts, and all 806 results are fully proved with zero `sorry`'s. Despite the definitional issues noted above, no proved theorem is *wrong*: the explicit assumptions compensate for the weak definitions at every point of use.

Proof structure. The extreme decomposition into helper lemmas—while verbose—produces proofs that are individually short, independently verifiable, and easy to debug. The main theorems (Tychonoff, Urysohn, Tietze, Nagata–Smirnov, Baire) read as clean assemblies of pre-verified components, making their logical structure transparent in a way that monolithic proofs do not.

Difficult results. The successful formalization of the nowhere-differentiable function (Theorem 49.1), which requires a delicate interplay of topology, real analysis, and uniform approximation, and of the Nagata–Smirnov metrization theorem (Theorem 40.3), with its 38 helper lemmas and intricate metric construction, demonstrates that LLM-assisted formalization can handle more than routine textbook exercises. The final push (March 30 – April 3) proved several further challenging results: the space-filling curve (Theorem 44.1), requiring a Hilbert-curve-style recursive construction with snake-order cell traversal and continuity of the limit function via uniform convergence; the dimension-theory imbedding theorems (Theorems 50.5 and 50.6), involving general-position arguments, partitions of unity, and weighted-sum constructions into $\mathbb{R}^{2m+1}$; and Corollary 45.5, characterizing compact subsets of complete metric spaces as closed and bounded—the very last sorry to be eliminated.

Source traceability. Definitions and theorems are annotated with their section and line number (`(** from §`n `... [top1.tex:`k`] **)`) in the source LaTeX, providing a complete correspondence between the informal and formal texts.

8.7 Summary

The qualitative review reveals a formalization that is *correct but rough*: the theorems are proved and the mathematics is sound, but the definitions carry unnecessary weaknesses, some theorem statements are incomplete relative to Munkres, proofs are unpolished, and the integration with Isabelle's existing infrastructure is ad hoc. These shortcomings are largely systematic—they reflect the LLM's consistent failure modes (defaulting to weak definitions, accepting verbose proofs without compression, adding redundant assumptions) rather than isolated errors. A focused cleanup pass, addressing the definitional issues and compressing proofs, would substantially increase the formalization's value as a reusable library; we estimate this would require days rather than weeks of expert effort, building on the sound foundation that the LLM-assisted workflow has established. Obviously, our analysis of the failure modes done here will also serve for further evolution of the rules given to the LLM agents in future autoformalization experiments.

9 Related Work

The first attempts at automated formalization (also known as semantic parsing of math) go back to the work Abrahams, Bobrow and McCarthy [4,1], followed later e.g. by Simon [20], Zinn [26] and Ganesalingam [7].[13] Modern learning-

[13] A brief talk about the history of autoformalization is available at `https://youtu.be/4JeezEGc_gQ`.

assisted autoformalization has evolved through several paradigms. Early approaches used probabilistic parsing via PCFGs combined with type checking and ATPs/hammers [12,13]. In 2018, Wang et al. [23] conducted first experiments with attention-based neural translation between LaTeX and Mizar, showing that the structural correspondence between (synthetic) informal and formal mathematics could be captured quite well by neural language models. Several deep-learning groups have followed this line of research since 2020 [25,11]. The current wave of direct LLM agent use began in late 2025 [22] and has accelerated rapidly. For instance, frontier LLMs with RL-based proof autocompletion has been used to verify the correctness of novel mathematical results in Lean [10].

Within Isabelle, the existing topology library (based on type classes [9]) takes a complementary approach; reconciling the two representations is an obvious future project. The Mizar Mathematical Library has had a large number of topology results since its early days, while in HOL Light, a lot of topology has been formalized more than a decade ago by Harrison. This is being considerably extended by Claude Code and Harrison today. Lean's Mathlib contains extensive topology formalized over several years by a large community. The present work's 24-day timeline for 85k lines invites comparison with these multi-year efforts, though the level of polish and library integration admittedly differs.

10 Conclusion

We have described an experiment that produced 85,472 lines of formal topology in Isabelle/HOL over 24 active days, covering all 39 sections of Munkres with 806 formal results and zero remaining gaps. The sorry-first methodology, combined with bulk sledgehammer invocations and disciplined build-time management, appears to provide an effective framework for LLM-assisted formalization in Isabelle. Further work could explore the LLM's ability to complete the textbook's exercises and perform the post-cleanup addressing the issues raised in Section 8.

The session log analysis reveals that the interaction with the LLM agent is approximately 83% automated, with human interventions primarily serving to correct the LLM's tendency toward tactic explosion, cherry-picking easy goals, and other behaviors commonly observed in iterative human–machine workflows.

As the fourth author noted in the Megalodon paper: "It is quite possible that in 2026 we will get most of (reasonably written) math textbooks and papers autoformalized." The present work, carried out two months later with a *complete* formalization in a different system, supports this assessment.

11 Acknowledgments

This work was supported by the ERC project NextReason, Amazon Research Awards, Renaissance Philanthropy grant DEEPER, and the sponsors of the AI4REASON institute.[14]

[14] `https://ai4reason.eu/sponsors.html`

We would like to thank Larry Paulson for his influential work on Isabelle, Sledgehammer, and formalization, including his recent interest in autoformalization. Many people—including some of us—have benefited over the years from Larry's expertise, scientific rigor, intellectual honesty, and generosity in discussing, testing, and supporting new ideas. His papers and talks have been a source of both insight and enjoyment, often combining deep technical contributions with a healthy dose of humor.[15] Our own interactions with Larry, including a shared interest in hammers [3], translations, and premise selection, contributed to early systems connecting automated reasoning, machine learning, and interactive theorem proving—and left us with some good stories to tell.[16]

References

1. P. W. Abrahams: Machine verification of mathematical proof. Sc.D. thesis, Massachusetts Institute of Technology (1966)
2. G. Bancerek et al. The role of the Mizar Mathematical Library for interactive proof development in Mizar. In: Journal of Automated Reasoning. Springer (2017).
3. J. C. Blanchette, C. Kaliszyk, L. C. Paulson, J. Urban: Hammering towards QED. *J. Formaliz. Reason.* 9(1), 101–148 (2016)
4. D. G. Bobrow: Natural language input for a computer problem solving system. PhD thesis, Massachusetts Institute of Technology (1964)
5. A. Bordg, L. C. Paulson, W. Li: Simple Type Theory is not too Simple: Grothendieck's Schemes Without Dependent Types In: *Exp. Math.* 31(2), 364–382 (2022)
6. C. E. Brown, C. Kaliszyk, M. Suda, J. Urban: Hammering higher order set theory. In: CICM 2025, LNCS 16136, pp. 3–20. Springer (2025)
7. M. Ganesalingam: The language of mathematics: A linguistic and philosophical investigation. LNCS 7805. Springer, Berlin Heidelberg (2013)
8. A. Hahn, A. De Lon: Understandable Autoformalization with Felix. Submitted to ITP 2026
9. J. Hölzl, F. Immler, B. Huffman: Type Classes and Filters for Mathematical Analysis in Isabelle/HOL. In: ITP 2013, LNCS, pp. 279–294. Springer (2013)
10. V. Ilin: Semi-Autonomous Formalization of the Vlasov-Maxwell-Landau Equilibrium. arXiv:2603.15929v2 (2026)
11. A. Q. Jiang, S. Welleck, J. P. Zhou, T. Lacroix, J. Liu, W. Li, M. Jamnik, G. Lample, Y. Wu: Draft, Sketch, and Prove: Guiding Formal Theorem Provers with Informal Proofs. In: ICLR 2023, Kigali, Rwanda, May 1–5. OpenReview.net (2023)
12. C. Kaliszyk, J. Urban, J. Vyskocil, H. Geuvers: Developing corpus-based translation methods between informal and formal mathematics. In: CICM 2014, LNCS 8543, pp. 435–439. Springer (2014)
13. C. Kaliszyk, J. Urban, J. Vyskocil: Learning to parse on aligned corpora (rough diamond). In: ITP 2015, LNCS 9236, pp. 227–233. Springer (2015)
14. D. Mulligan, L. C. Paulson: Isabelle/Copilot: An AI-driven assistant for Isabelle/HOL (2026). `https://lists.cam.ac.uk/sympa/arc/cl-isabelle-users/2026-02/msg00039.html`
15. H. Becker, D. Mulligan, et. al: Isabelle/Assistant (2026). `https://github.com/awslabs/AutoCorrode/tree/main/isabelle-assistant`

[15] `https://www.youtube.com/watch?v=-Eewz-DJovU&t=2414s`
[16] `https://www.youtube.com/watch?v=vUlDvBzagK4&t=1442s`

16. J. R. Munkres: *Topology*, 2nd edn. Prentice Hall (2000)
17. T. Nipkow, L. C. Paulson, M. Wenzel: *Isabelle/HOL—A Proof Assistant for Higher-Order Logic*. LNCS 2283. Springer (2002)
18. L.C. Paulson: Isabelle: The Next Seven Hundred Theorem Provers. In: CADE 1988, LNCS, pp. 772–773. Springer (1988)
19. L. Paulson: Porting the HOL Light analysis library: some lessons. In: CPP 2017, pp. 1. ACM (2017)
20. D. Simon: Checking Number Theory Proofs in Natural Language. Ph. D. Dissertation. University of Texas at Austin. (1990)
21. G. Sutcliffe: The TPTP Problem Library and Associated Infrastructure - From CNF to TH0. In: J. Autom. Reason., pp. 483–502. Springer (2017)
22. J. Urban: 130k Lines of Formal Topology in Two Weeks: Simple and Cheap Autoformalization for Everyone? arXiv:2601.03298 (2026)
23. Q. Wang, C. Kaliszyk, J. Urban: First experiments with neural translation of informal to formal mathematics. In: CICM 2018, LNCS 11006, pp. 255–270. Springer (2018)
24. D. Williams: *Probability with Martingales*. Cambridge: Cambridge University Press. (1991)
25. Y. Wu, A. Q. Jiang, W. Li, M. N. Rabe, C. Staats, M. Jamnik, C. Szegedy: Autoformalization with Large Language Models. In: Advances in Neural Information Processing Systems 35 (NeurIPS 2022), New Orleans, LA, USA, November 28–December 9. (2022)
26. C. Zinn: Understanding informal mathematical discourse. PhD thesis, Institut für Informatik, Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany (2004)