Herald: A Natural Language Annotated Lean 4 Dataset

Guoxiong Gao, Yutong Wang, Jiedong Jiang, Qi Gao, Zihan Qin, Tianyi Xu, Bin Dong

Speaker: Jiedong Jiang

ICLR 2025, Singapore EXPO

Outline

- Background
- Herald Dataset
- Herald Translator
- Future Work

Background

Overview of interactive theorem prover and formalization in Lean

What is formal language (ITP) ?

- Interactive Theorem Provers:
  - build theorems from axioms
  - show proof states
  - **verify** the proof
- Lean is a popular ITP
- Mathlib4 is its mathematical library

Tactic

```
theorem Group.class_equation [Fintype G]:
  card (Subgroup.center G) + ∑ x ∈ noncenter G, card x.carrier = card G := by
  /- Rewrite `G` as partitioned by its conjugacy classes -/
  nth_rw 2 [← sum_conjClasses_card_eq_card']
  /- Cancel out nontrivial conjugacy classes from summation -/
  rw [← Finset.sum_sdiff (ConjClasses.noncenter G).toFinset.subset_univ]; congr 1
  /- Now we can obtain the result by calculation -/
  calc
    _ = card ((noncenter G)ᶜ : Set (ConjClasses G)) :=
      card_congr ((mk_bijOn G).equiv _)
    _ = Finset.card (Finset.univ \ (noncenter G).toFinset) := by
      rw [← Set.toFinset_card, Set.toFinset_compl, Finset.compl_eq_univ_sdiff]
    _ = ∑ x ∈ Finset.univ \ (noncenter G).toFinset, 1 :=
      Finset.card_eq_sum_ones _
    _ = ∑ x ∈ Finset.univ \ (noncenter G).toFinset, card x.carrier := by
      rw [Finset.sum_congr rfl _];
      rintro ⟨g⟩ hg; simp at hg
      rw [← Set.toFinset_card, eq_comm, Finset.card_eq_one]
      exact ⟨g, by
        rw [← Set.toFinset_singleton];
        exact Set.toFinset_congr (Set.Subsingleton.eq_singleton_of_mem hg mem_carrier_mk)⟩
```

Formalization

Theorem 7. *(The Class Equation)* Let G be a finite group and let $g_1, g_2, \ldots, g_r$ be representatives of the distinct conjugacy classes of G not contained in the center $Z(G)$ of G. Then

$$|G| = |Z(G)| + \sum_{i=1}^{r} |G : C_G(g_i)|.$$

Proof: As noted in Example 2 above the element $\{x\}$ is a conjugacy class of size 1 if and only if $x \in Z(G)$, since then $gxg^{-1} = x$ for all $g \in G$. Let $Z(G) = \{1, z_2, \ldots, z_m\}$, let $\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_r$ be the conjugacy classes of G not contained in the center, and let g_i be a representative of $\mathcal{K}_i$ for each i. Then the full set of conjugacy classes of G is given by

$$\{1\},\ \{z_2\},\ \ldots,\ \{z_m\},\ \mathcal{K}_1,\ \mathcal{K}_2,\ \ldots,\ \mathcal{K}_r.$$

Since these partition G we have

$$\begin{aligned} |G| &= \sum_{i=1}^{m} 1 + \sum_{i=1}^{r} |\mathcal{K}_i| \\ &= |Z(G)| + \sum_{i=1}^{r} |G : C_G(g_i)|, \end{aligned}$$

Natural language version

25x slower

2x-10x longer

```
theorem Group.class_equation [Fintype G]:
  card (Subgroup.center G) + ∑ x ∈ noncenter G, card x.carrier = card G := by
  /- Rewrite `G` as partitioned by its conjugacy classes -/
  nth_rw 2 [← sum_conjClasses_card_eq_card']
  /- Cancel out nontrivial conjugacy classes from summation -/
  rw [← Finset.sum_sdiff (ConjClasses.noncenter G).toFinset.subset_univ]; congr 1
  /- Now we can obtain the result by calculation -/
  calc
    _ = card ((noncenter G)ᶜ : Set (ConjClasses G)) :=
      card_congr ((mk_bijOn G).equiv _)
    _ = Finset.card (Finset.univ \ (noncenter G).toFinset) := by
      rw [← Set.toFinset_card, Set.toFinset_compl, Finset.compl_eq_univ_sdiff]
    _ = ∑ x ∈ Finset.univ \ (noncenter G).toFinset, 1 :=
      Finset.card_eq_sum_ones _
    _ = ∑ x ∈ Finset.univ \ (noncenter G).toFinset, card x.carrier := by
      rw [Finset.sum_congr rfl _];
      rintro ⟨g⟩ hg; simp at hg
      rw [← Set.toFinset_card, eq_comm, Finset.card_eq_one]
      exact ⟨g, by
        rw [← Set.toFinset_singleton];
        exact Set.toFinset_congr (Set.Subsingleton.eq_singleton_of_mem hg mem_carrier_mk)⟩
```

Formal language version

Translation

Stone-Weierstrass theorem
A fundamental result in real analysis stating that any continuous function…

```
theorem exists_polynomial
(a b : ℝ) (f : C(Set.Icc a b, ℝ))
(ε : ℝ)(pos : 0 < ε) : ∃ p :ℝ[X],
‖p.toContinuousMapOn _ - f‖ < ε
```

Natural language statement

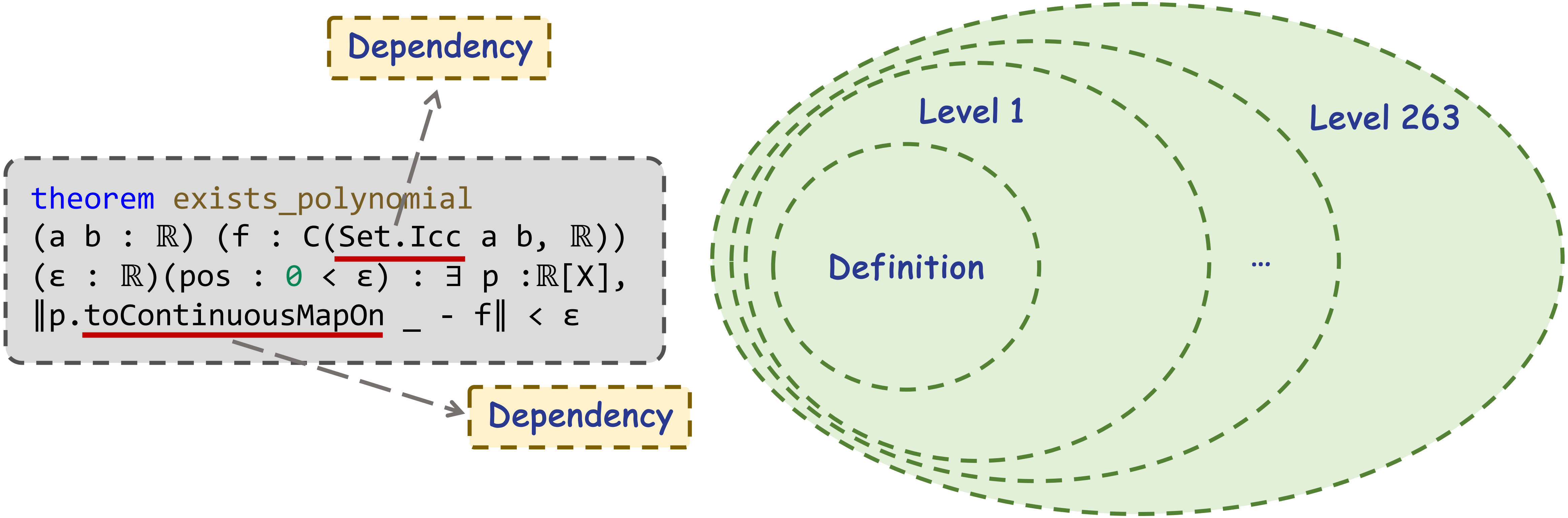
Formal language statement

How can we train LLM for this?

Herald Dataset

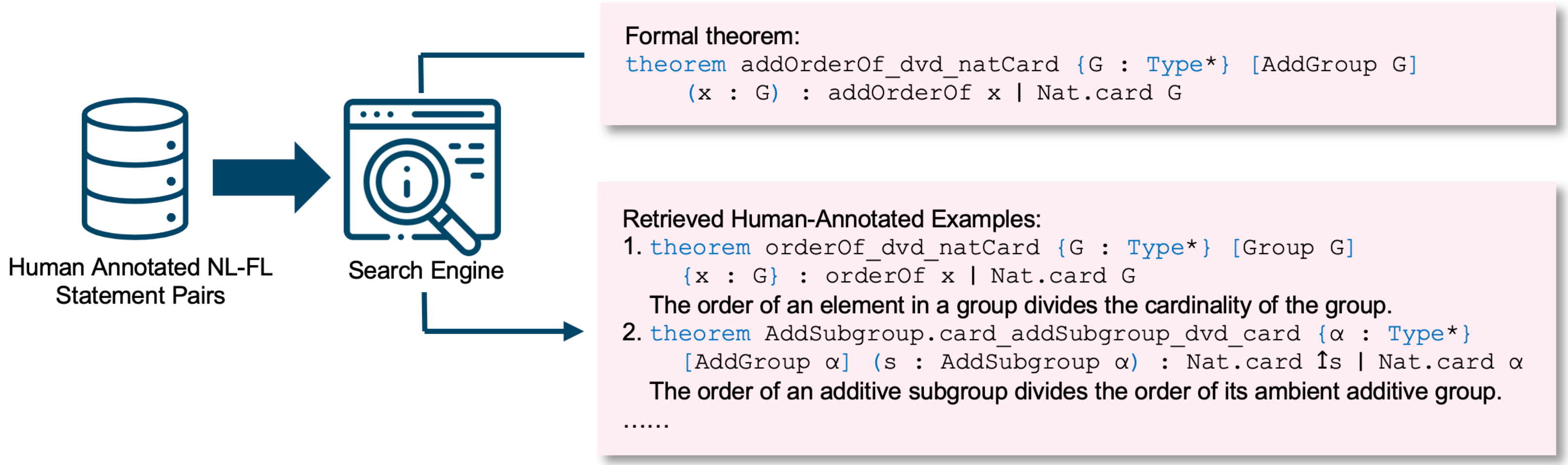
Pipelines for auto-informalization and augmentations

Dependency hierarchy



Retrieval-Augmented Generation

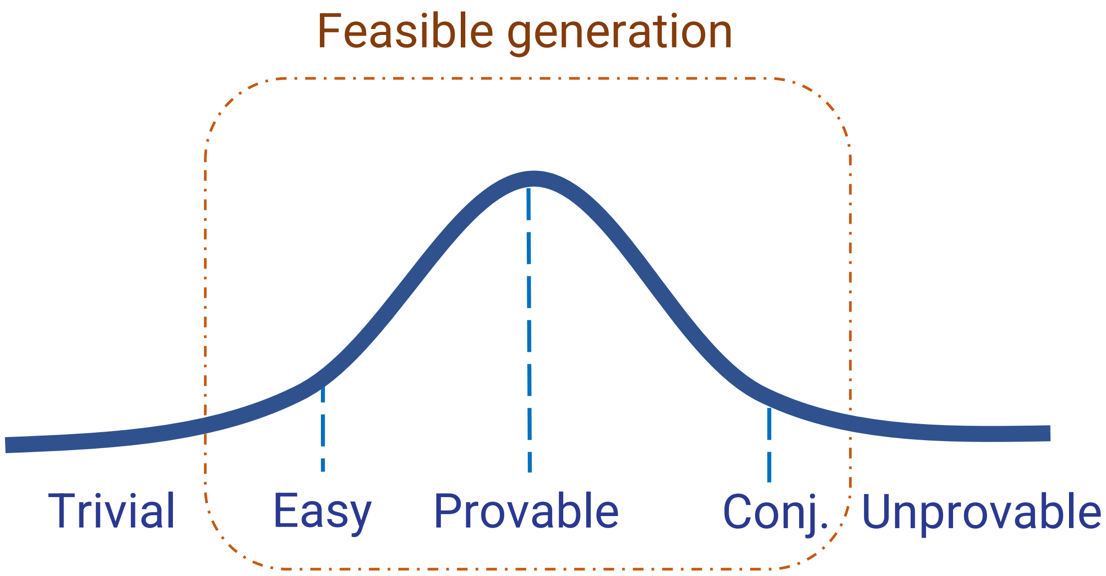
- Contextual information
- Retrieval human-annotated similar examples use LeanSearch



Gao, Guoxiong, et al. A Semantic Search Engine for Mathlib4. EMNLP, 2024.

Augmentation

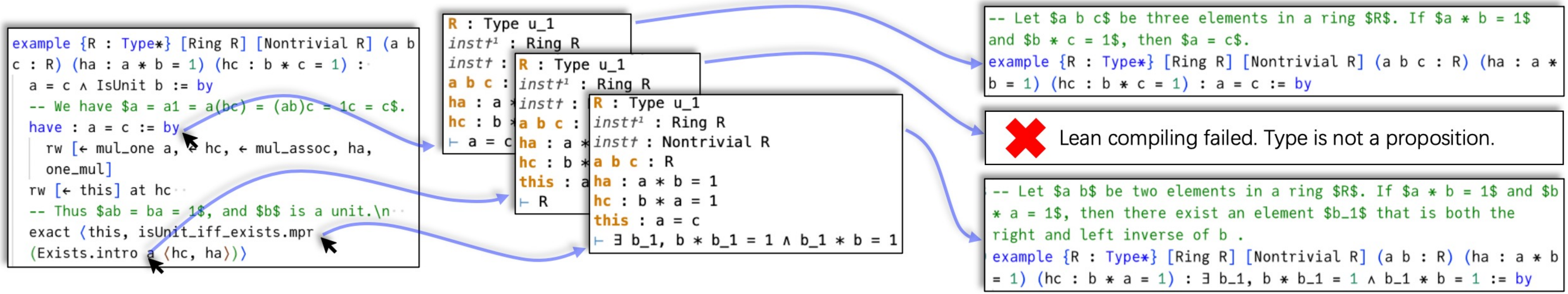
- What if data is still not enough?
- Previous efforts:
  - Symbolic generator
  - Swap conditions
- Swap lead to repetitions
- Random generation collapse the curve
- We need **provable data**



Distribution of Theorem in Real World

Augmentation

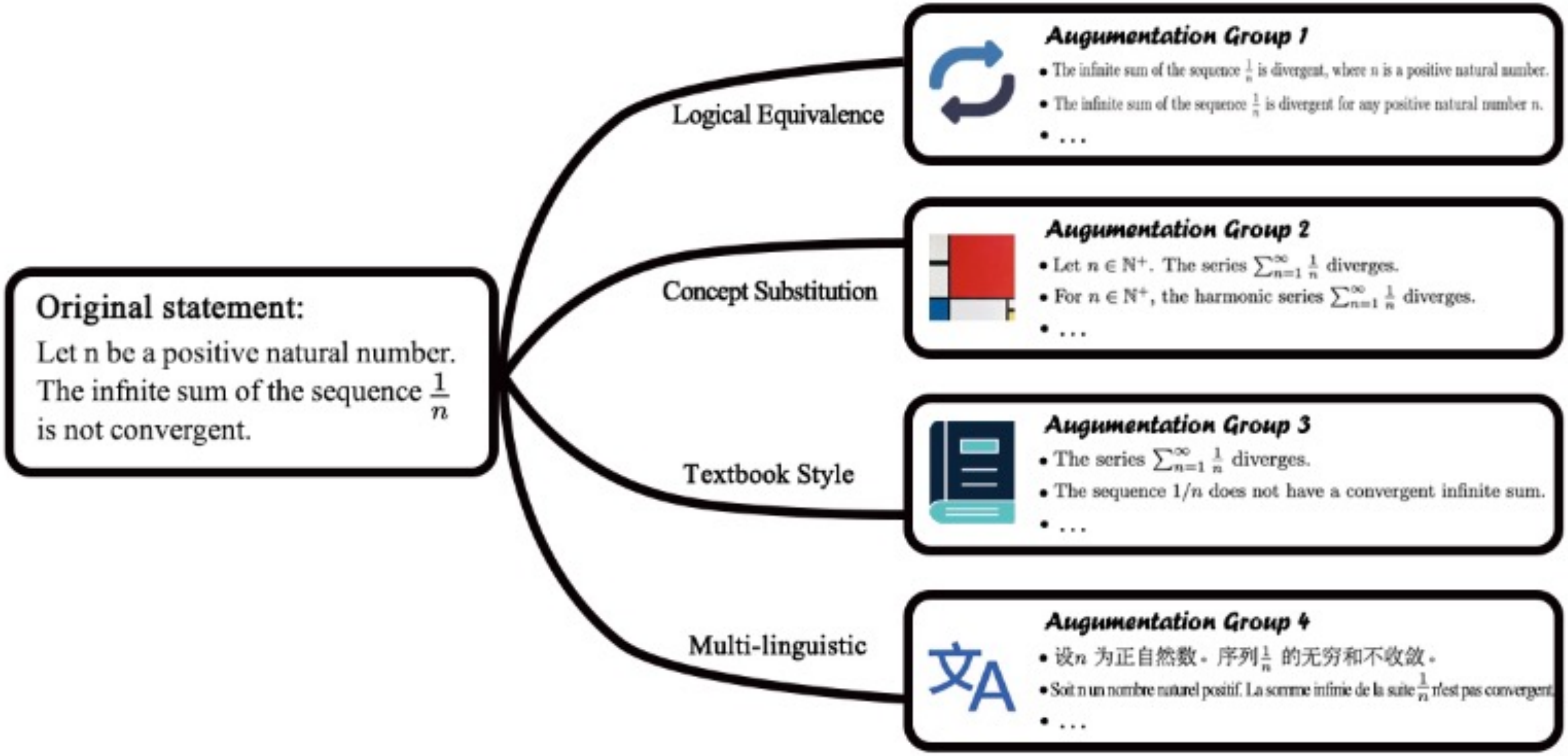
- Our approach: Use proof state
- Advantage: All local state are provable.



Formal Proof | Extracted Proof State | Validated NL-FL Statement Pair

https://github.com/reaslab/jixia

Augmentation



Augmentation Result

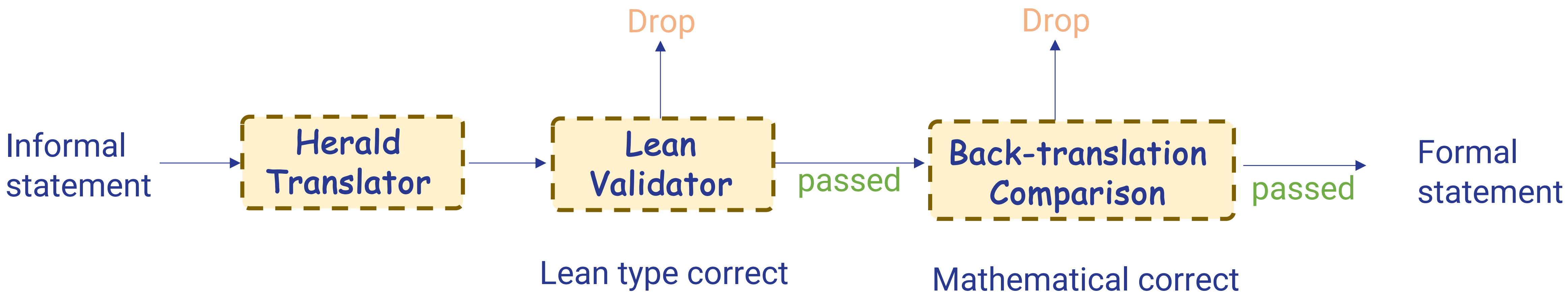
Obtaining 580k statements out of 291k raw statement in Mathlib4.

	Mathlib4 Original Statements	Augmented Statements	Mathlib4 Proofs
Number of NL-FL pairs	$291k$	$580k$	$44k$

Herald Translator

Auto-formalization of mathematical statements

Inferencing



Result

Model	miniF2F		Extract Theorem	College CoT
	test	valid		
TheoremLlama	50.1%	55.6%	4.0%	2.9%
InternLM2-Math-Plus-7B	73.0%	80.1%	7.5%	6.5%
Llama3-instruct	28.2%	31.6%	3.6%	1.8%
Herald	**93.2%**	**96.7%**	**22.5%**	**17.1%**

Table 2: Performance comparison of different models across various datasets. The last two datasets (Extract Theorem and College CoT) are shuffled subsets of 200 samples each.

Real-world Formalization Project

- Formalized Stacks Project (online resource of algebraic geometry and related topics), section Normal Extensions.

```lean
import Mathlib

open Polynomial

/-- Let $K / E / F$ be a tower of algebraic field extensions. If $K$ is
    normal over $F$, then $K$ is normal over $E$.-/
theorem tower_top_of_normal (F E K : Type*) [Field F] [Field E]
    [Algebra F E]
[Field K] [Algebra F K] [Algebra E K] [IsScalarTower F E K] [h : Normal
    F K] :
Normal E K := by
  -- We use the fact that normality is equivalent to being a normal
    extension.
  have := h.out
  -- The above statement is a direct consequence of the transitivity of
    normality.
  exact Normal.tower_top_of_normal F E K

/-- Let $F$ be a field. Let $M / F$ be an algebraic extension. Let $M /
    E_i / F$, $i \in I$ be subextensions with $E_i / F$ normal. Then $
    \bigcap E_i$ is normal over $F$.-/
theorem normal_iInf_of_normal_extracted {F M : Type*} [Field F] [Field
    M] [Algebra F M] {E : ι → IntermediateField F M}
[Algebra.IsAlgebraic F M] : (∀ (i : ι), Normal F ↥(E i)) → Normal F
    ↥⨅( i, E i) := by sorry
```

```lean
/-- Let $E / F$ be an algebraic field extension. Let $E / F$ be a
    normal algebraic field extension. There exists a unique
    subextension $E / E_{ ext {sep }} / F$ such that $E_{ ext {sep }} /
    F$ is separable and $E / E_{ ext {sep }}$ is purely inseparable.
    The subextension $E / E_{ ext {sep }} / F$ is normal. -/
theorem normal_ext_sep_ext'_ext_tac_28642 [Field F] [Field E] [Algebra
    F E] [Algebra.IsAlgebraic F E] (h : Normal F E) (this : Algebra
    ↥(separableClosure F E) E) : Normal ↥(separableClosure F E) E :=
    by sorry

/-- Let $E / F$ be an algebraic extension of fields. Let $\bar{F}$ be
    an algebraic closure of $F$. The following are equivalent
(1)  $E$ is normal over $F$, and
(2)  for every pair $\sigma, \sigma^{\prime} \in \operatorname{Mor}_F(E,
    \bar{F})$ we have $\sigma(E)=\sigma^{\prime}(E)$. -/
theorem normal_iff_forall_map_eq_of_isAlgebraic_ext_ext {F E : Type*}
    [Field F]
[Field E] [Algebra F E] [Algebra.IsAlgebraic F E] (overlineF : Type*)
    [Field overlineF]
[Algebra F overlineF] [IsAlgClosure F overlineF] :
Normal F E ↔ ∀ (σ σ' : E →ₐ[F] overlineF), Set.range ⇑σ = Set.range ⇑σ
    ' := by
sorry
```

Future Work

How far can we push in data preparation?

Future Work

- How can we further assist human experts in formalization?
- How far can we push by augmenting and using the existing dataset?

Thank you!