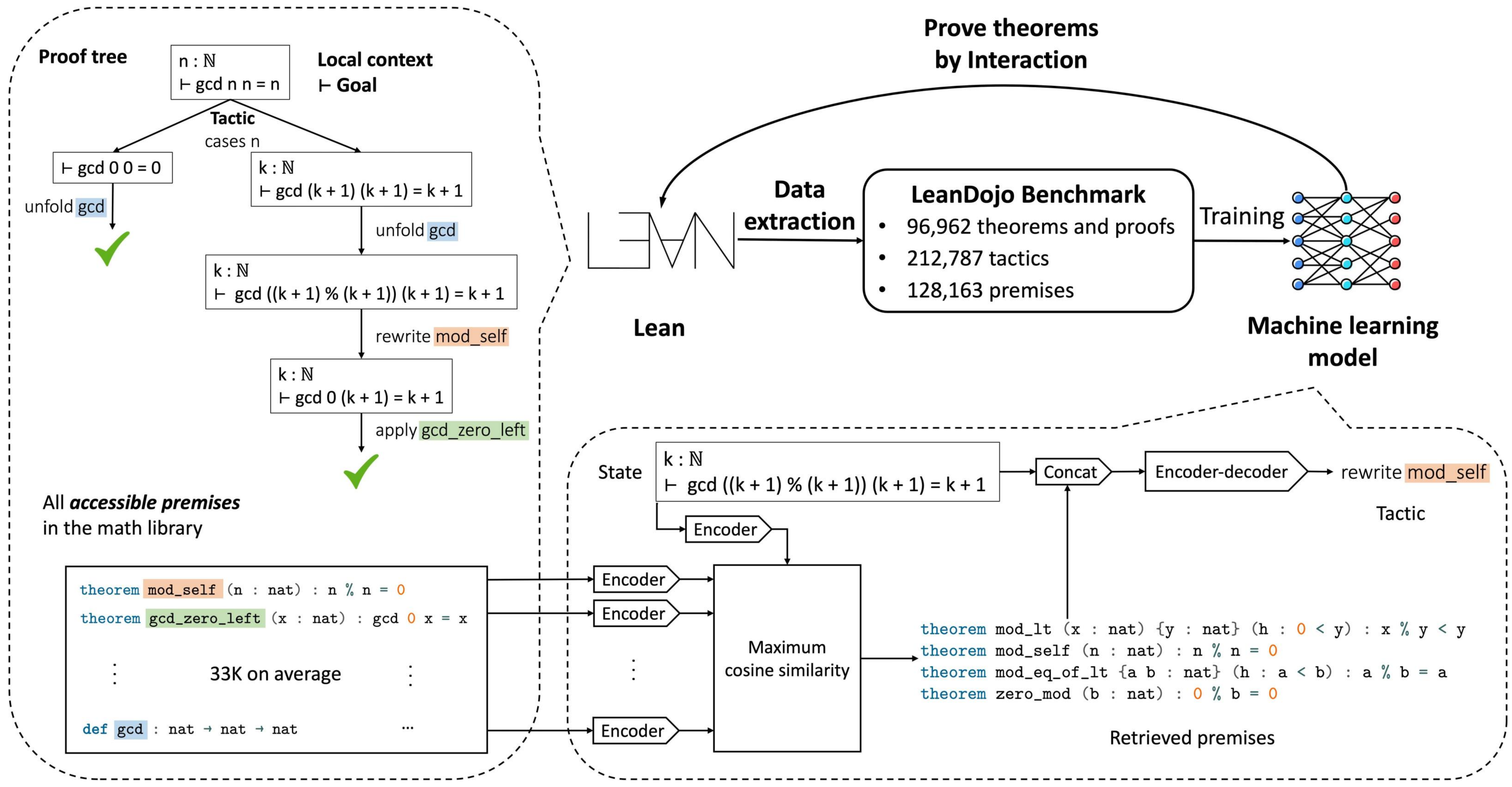
LeanDojo: Theorem Proving with Retrieval Augmented Language Models

Kaiyu Yang[1], Aidan Swope[2], Alex Gu[3], Rahul Chalamala[1], Peiyang Song[4], Shixing Yu[5], Saad Godil[2], Ryan Prenger[2], Anima Anandkumar[1 2]

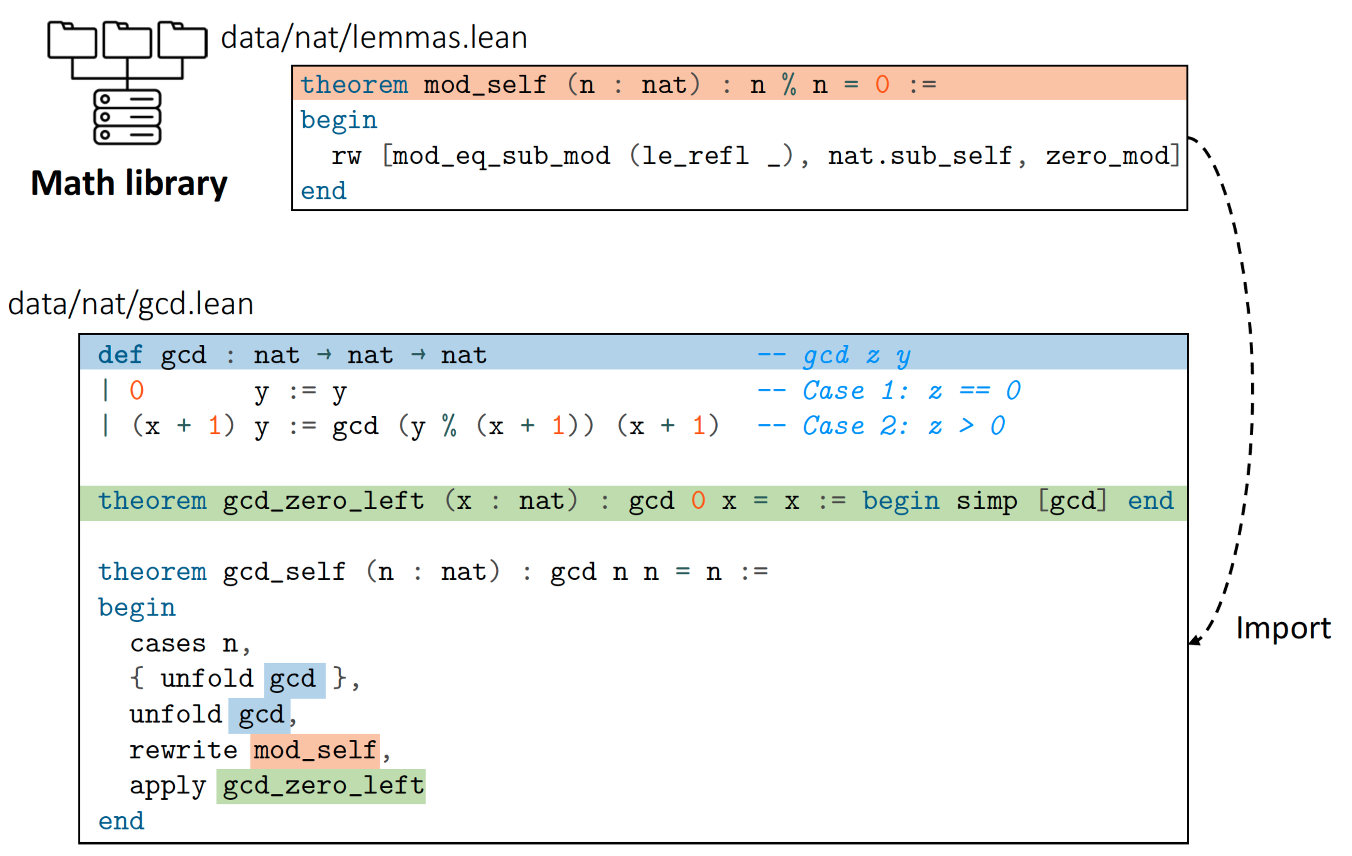
[1]Caltech; [2]NVIDIA; [3]MIT; [4]UC Santa Barbara; [5]UT Austin



Top right: LeanDojo extracts proofs in Lean into datasets for training machine learning models. It also enables the trained model to prove theorems by interacting with Lean's proof environment.

Top left: The proof tree of a Lean theorem $\forall n \in \mathbb{N}$, `gcd n n = n`, where `gcd` is the greatest common divisor. When proving the theorem, we start from the original theorem as the initial state (the root) and repeatedly apply tactics (the edges) to decompose states into simpler sub-states, until all states are solved (the leaf nodes). Tactics may rely on premises such as `mod_self` and `gcd_zero_left` defined in a large math library. E.g., `mod_self` is an existing theorem $\forall n \in \mathbb{N}$, `n % n = 0` used in the proof to simplify the goal.

Bottom: Our ReProver model. Given a state, it retrieves premises from the math library, which are concatenated with the state and fed into an encoder-decoder Transformer to generate the next tactic.



Formalizing and proving theorems in Lean

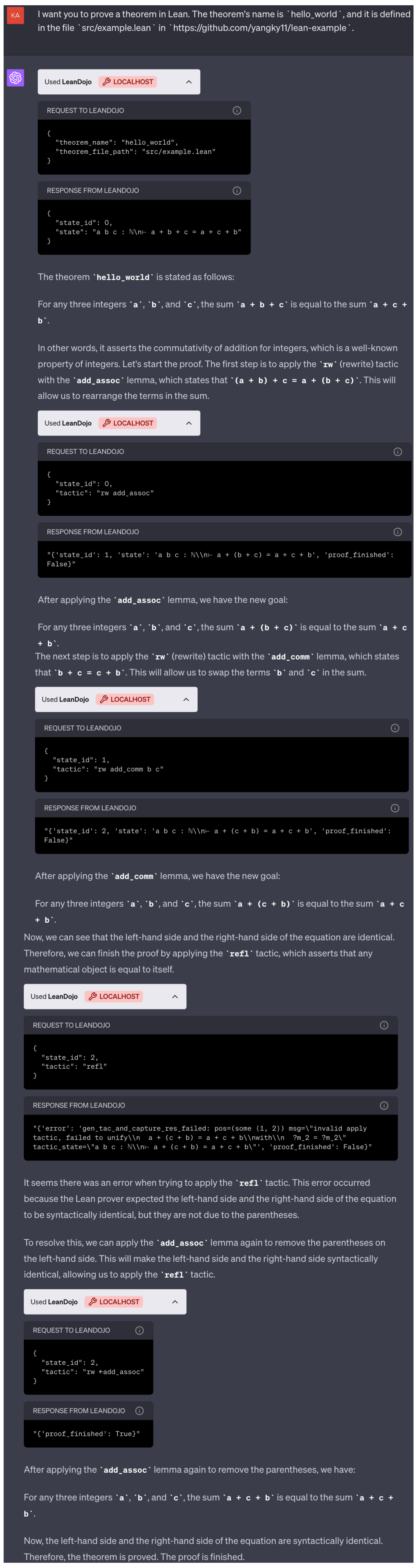
Method	random	novel_premises
tidy	23.8	5.4
GPT-4	28.8	7.5
ReProver (ours)	**51.4**	**26.2**
w/o retrieval	47.5	22.9

% of theorems proved within 10 minutes

```
theorem exercise_2_3_2 {G : Type*} [group G] (a b : G) :
   g : G, b * a = g * a * b * g ¹ :=
begin
  exact  b, by simp ,
end

theorem exercise_11_2_13 (a b :  ) :
  (of_int a : gaussian_int)   of_int b → a   b :=
begin
  contrapose,
  simp,
end

theorem exercise_1_1_17 {G : Type*} [group G] {x : G} {n :  }
  (hxn: order_of x = n) :
  x ¹ = x ^ (n - 1 :  ) :=
begin
  rw zpow_sub_one,
  simp,
  rw [← hxn, pow_order_of_eq_one],
end

theorem exercise_3_1_22b {G : Type*} [group G] (I : Type*)
  (H : I → subgroup G) (hH :   i : I, subgroup.normal (H i)) :
  subgroup.normal (  (i : I), H i):=
begin
  rw infi,
  rw ←set.image_univ,
  rw Inf_image,
  simp [hH],
  haveI :=   i, (H i).normal,
  split,
  intros x hx g,
  rw subgroup.mem_infi at hx  ,
  intro i,
  apply (hH i).conj_mem _ (hx i),
end

theorem exercise_3_4_5a {G : Type*} [group G]
  (H : subgroup G) [is_solvable G] : is_solvable H :=
begin
  apply_instance,
end
```

New proofs discovered by ReProver

The first open-source LLMs for theorem proving



ChatGPT proves theorems by interacting with Lean