Learning to Reason and Align via Self-Play

In pursuit of Superhuman Intelligence

Ziyu Ye

Dec 06, 2024

THE UNIVERSITY OF CHICAGO

Research Goal: *In Pursuit of Superintelligence*

~ Self-Play

Design scalable methods for intelligence to perform complex sequential decision making to achieve goals in the open world.

~ Reasoning

~ Alignment

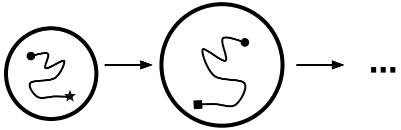
Agenda for Today's Talk

•

Intro: Learning as an Game

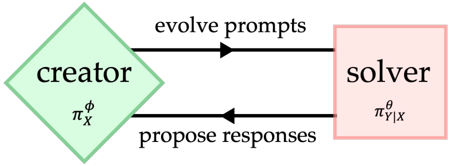
Conventional Training as a Finite Game

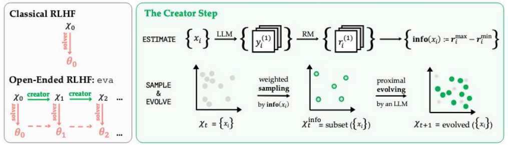
New, Scalable Training as an Infinite Game



• •

Self-Play to Align





Algorithm 1 eva: Evolving Alignment *via* Asymmetric Self-Play

Input: initial policy π_{θ_0}, initial prompt set $\mathcal{X}_0$

1: **for** iteration $t = 1, 2, \ldots$ **do**

▷ /* `creator step` */

2: *estimate informativeness:* $\mathcal{X}_{t-1} \leftarrow \{(\boldsymbol{x}_i, \texttt{info}(\boldsymbol{x}_i)) \mid \boldsymbol{x}_i \in \mathcal{X}_{t-1}\}$

sample subset: $\mathcal{X}_{t-1}^{\text{info}} \leftarrow \texttt{sample}(\mathcal{X}_{t-1})$

self-evolve prompts: $\mathcal{X}_t \leftarrow \texttt{evolve}(\mathcal{X}_{t-1}^{\text{info}})$

▷ /* `solver step` */

3: *self-generate responses:* $\forall \boldsymbol{x}_i \in \mathcal{X}_t$, generate $\{\boldsymbol{y}_i^{(j)}\} \sim \pi_{\theta_{t-1}}(\cdot \mid \boldsymbol{x}_i)$

annotate rewards: $\mathcal{X}_t' \leftarrow \mathcal{X}_t \cup \{(\boldsymbol{y}_i^{(j)}, r_i^{(j)})\}$

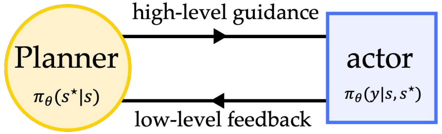
preference optimization: $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{X}_t'}(\boldsymbol{\theta})$

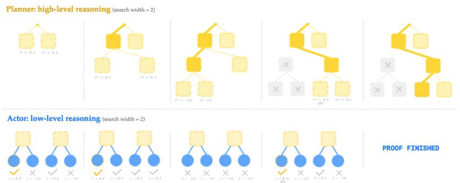
4: **end for**

5: **return** final solver policy π_{θ_T}

• • •

Self-Play to Reason





Algorithm 1 RiR – *A Unified Reasoning Mechanism with Decomposing and Search*

Input: problem statement $\boldsymbol{q}$, a language model w/ parameter $\boldsymbol{\theta}$

```
1: tree ← Tree(θ, q)
2: repeat
3:     s*_l ← tree.policy()                ▷ /* planner */
4:     tree ← Tree(θ, s*_l)
5:     repeat
6:         y_{t_l} ← tree.policy()         ▷ /* goal-driven actor */
7:     until STOP_LOW
8:     {tree, tree}.update()               ▷ /* joint update */
9: until STOP_HIGH
10: return tree.solution
```

Intro:
Learning as an Infinite Game

"a paradigm shift for training large models"

The Vision: *"Universality of Computation"*

6. *The universal computing machine.*

It is possible to invent a single machine which can be used to compute any computable sequence. If this machine $\mathfrak{U}$ is supplied with a tape on the beginning of which is written the S.D of some computing machine $\mathcal{M}$, then $\mathfrak{U}$ will compute the same sequence as $\mathcal{M}$.

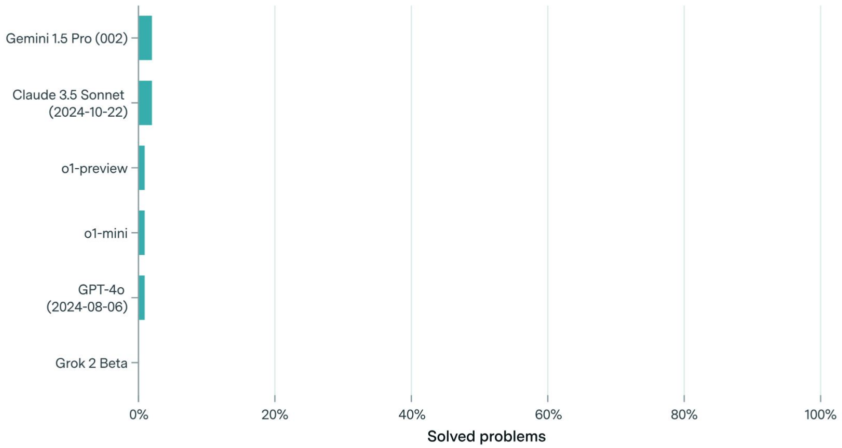
– Alan M. Turing, 1936

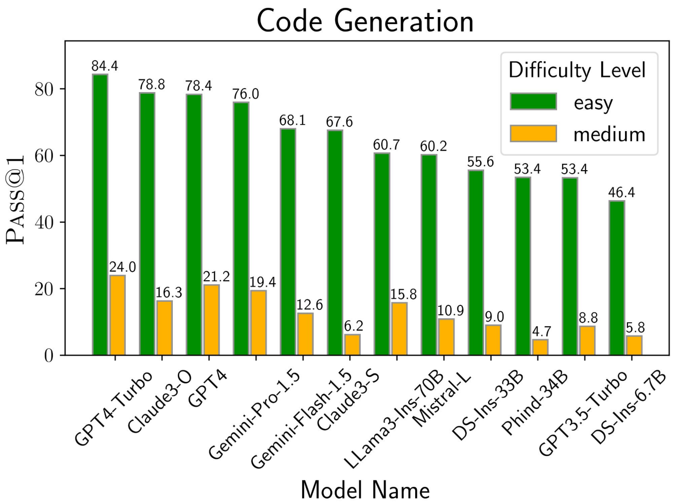
"what a *human* can think or know"

=

"what a *machine* can compute"

The Challenge: Gaps in Achieving Human-Level Performance





- **< 2% accuracy** on challenging contemporary mathematics problems on FrontierMath.
- **>3.5x performance drop** as coding problems get harder on LiveCodeBench.

What may go wrong in conventional ways of training AI models?

The Challenge: Scaling Law is Hitting the Wall? 🫢

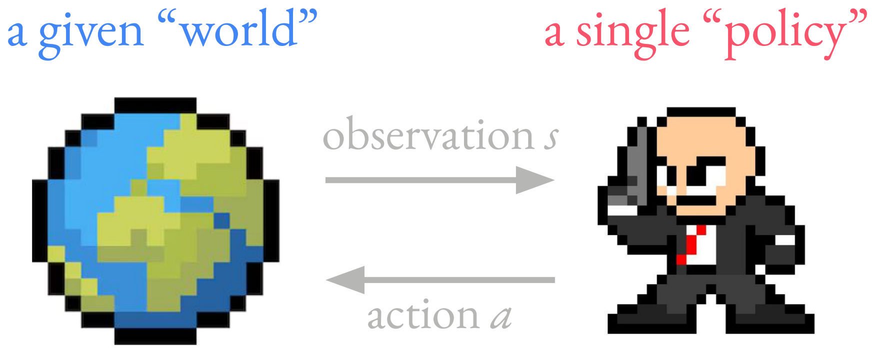
"Ilya Sutskever, co-founder of AI labs Safe Superintelligence (SSI) and OpenAI, told Reuters recently that results from scaling up pre-training - **the phase of training an AI model** that use s a vast amount of unlabeled data to understand language patterns and structures - **have plateaued**."

"'The 2010s were the age of scaling, now we're back in the age of wonder and discovery once again. Everyone is looking for **the next thing**,'" Sutskever said. "'**Scaling the right thing matters more now than ever.**'"

– Ilya Sutskever with Reuters, Nov 2024

What are the next right things to scale?

Conventional Way of Agent Training



- **Intelligence:** Agents that are able to *learn* to *make decisions* ▲ to *achieve goals* ■.
- ▲ **Reasoning:** The process of *making decisions* by evaluating information.
- ■ **Alignment:** The process of *achieving goals* by reward maximization.

Myth 1: Learning is Purely Solving (under a given world)

a given world

Conventional way:

Design agents that **find solutions** in a fixed environment, then **stop learning**.

the open-ended worlds

Better way:

Design agents that **create new tasks/environments**, then **continuously learn to self-improve**.

See also Abel et al., (2024) on the dogma of "Learning as Finding a Solution".

Myth 2: Reasoning is Step-by-Step (by a single policy)

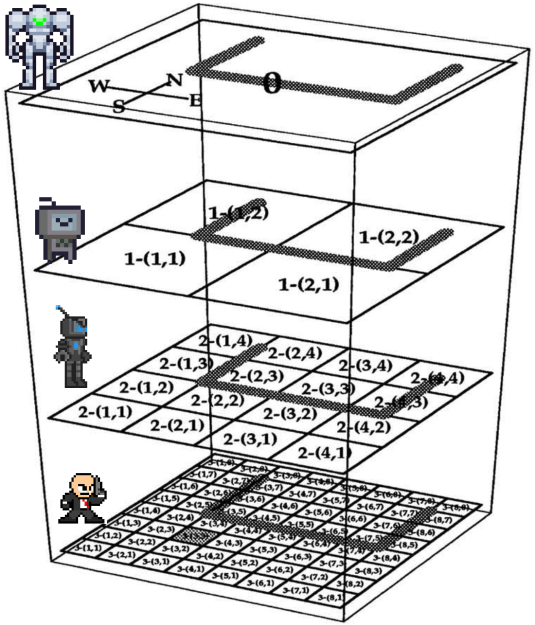


Fig 1. A world can be divided at different levels in certain hierarchy (Dayan and Hinton, 1992).

Better way:

Learn policies with **a hierarchy of abstract models**, and roll out at different levels for optimization.

Conventional way:

Learn a policy that operates under a **one-step model**, and roll it out (with tree search) in training.

See also Sutton (2024) on "The One-Step Trap in AI Research" and Yann LeCun's "Object-Driven AI with Hierarchical Planning".

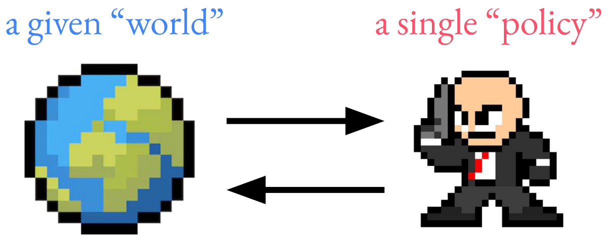
A New, Scalable Training Paradigm

"There are at least 2 kinds of games. One could be called finite; the other infinite."

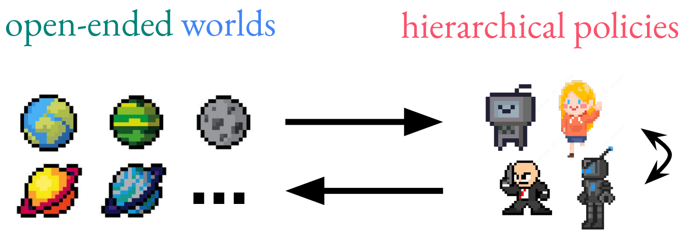
- A **finite game** is played for the purpose of winning.
- An **infinite game** is for the purpose of continuing the play.

– James P. Carse, 2011

Conventional Training as a Finite Game



New, Scalable Training as an Infinite Game



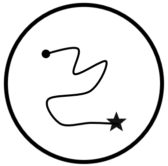
A New, Scalable Training Paradigm

"There are at least 2 kinds of games. One could be called finite; the other infinite."

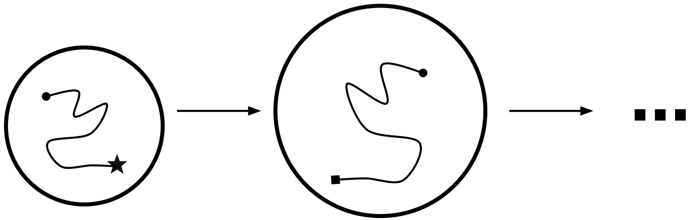
- A **finite game** is played for the purpose of winning.
- An **infinite game** is for the purpose of continuing the play.

– James P. Carse, 2011

Conventional Training as a Finite Game



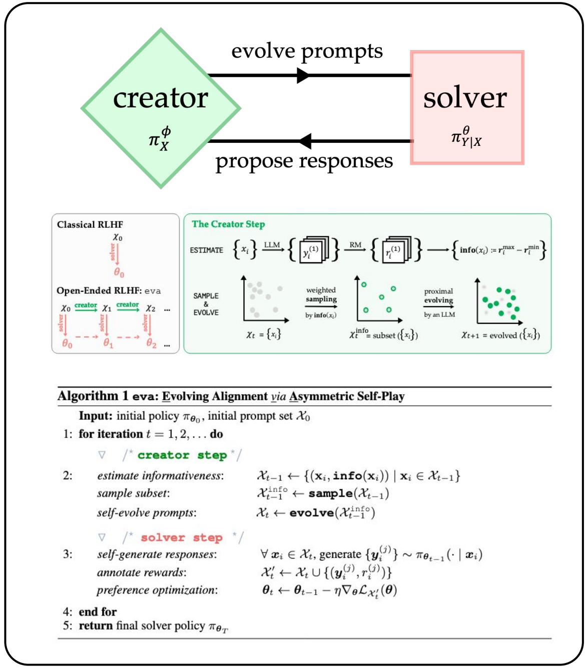
New, Scalable Training as a Infinite Game



This slide follows from and is inspired by Abel et al. (2023)'s "A Definition of Continual Reinforcement Learning".

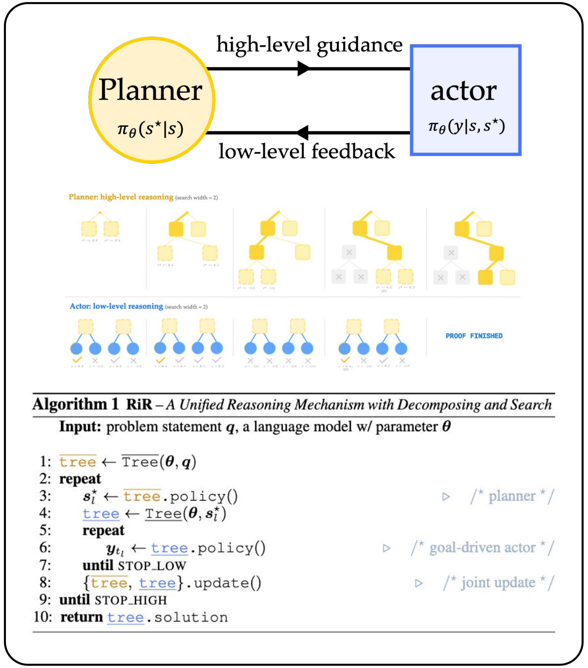
Recap on Agenda

Self-Play to Align



Solving Myth 1:
Going Beyond Static World

Self-Play to Reason



Solving Myth 2:
Hierarchical Planning

Self-Play to Align: The Creator-Solver Game

"Scalable language model training beyond human prompts."

Gratitude to every wonderful co-author of this project (link):
Rishabh Agarwal, Tianqi Liu, Rishabh Joshi, Sarmishta Velury, Quoc V. Le, Qijun Tan, Yuan Liu.

Google DeepMind
THE UNIVERSITY OF CHICAGO

TL; DR

We identify **learnable, worth-learning prompts** by **reward signals**, then **evolve new prompts** for open-ended continual RLHF training.

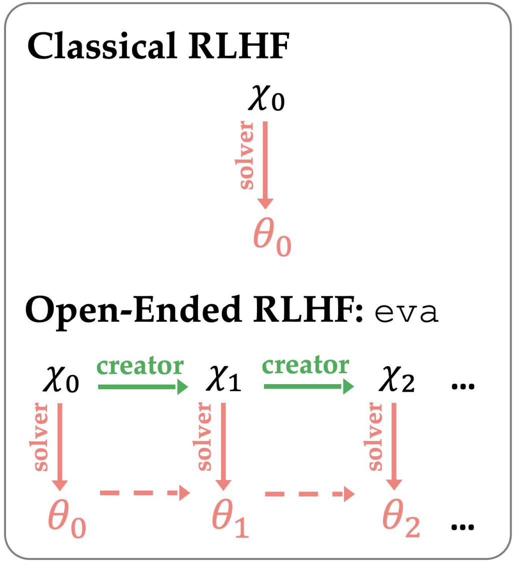


Fig 1. **RLHF needs a paradigm shift!**

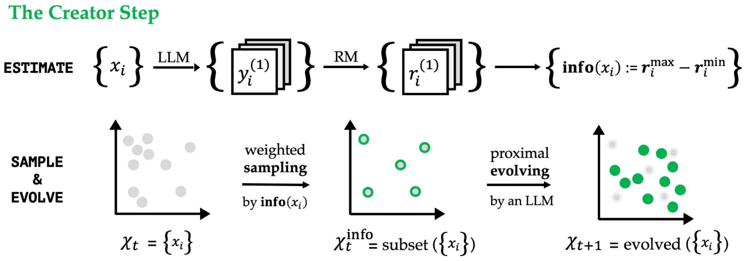


Fig 2. The **easy-to-implement pipeline** of **eva** for open-ended RLHF.

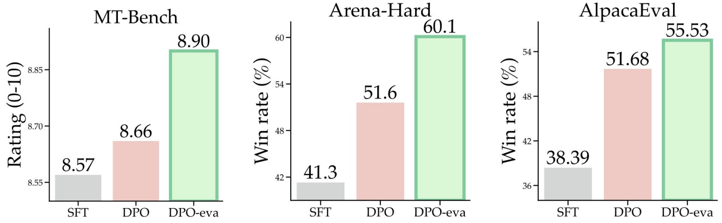


Fig 3. **eva** brings strong **alignment gain**.

Artificial Intelligence May Be Bottlenecked by Static Data

Reference: Villalobos et al., 2024

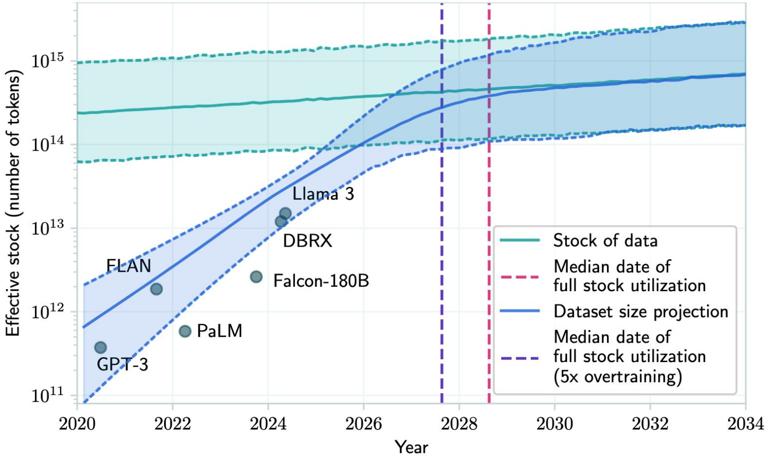


Fig 4. The **scale**, **quality** and **growth** of human knowledge is bottlenecked.

See also: lmsys-hard

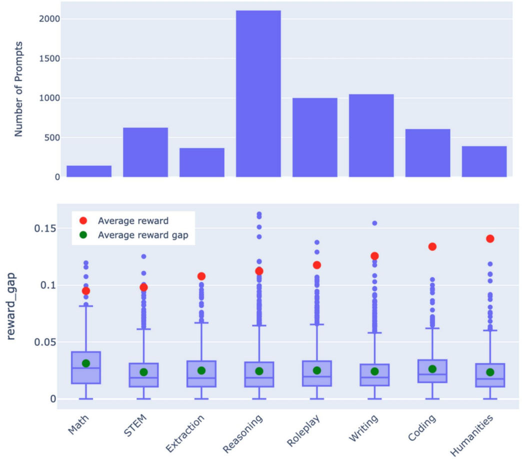


Fig 5. The **imbalance distribution** of static training data.

Can language models identify and self-create **new**, **learnable**, and **worth-learning** tasks, to self-improve to generalize better for alignment?

Classical RLHF

Alignment by RLHF. Classical RLHF (Ouyang et al., 2022) optimizes on a fixed distribution $\mathcal{D}$:

$$\max_{\pi_{\boldsymbol{\theta}}} \mathbb{E}_{\mathbf{x}\sim\mathcal{D},\mathbf{y}\sim\pi_{\boldsymbol{\theta}}(\cdot|\mathbf{x})}\Big[r(\mathbf{x},\mathbf{y})\Big] - \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\Big[\beta\cdot\mathbb{D}_{\mathrm{KL}}\big[\pi_{\boldsymbol{\theta}}(\mathbf{y}\mid\mathbf{x})\,\|\,\pi_{\mathrm{SFT}}(\mathbf{y}\mid\mathbf{x})\big]\Big], \tag{1}$$

where $\mathbf{x}$ and $\mathbf{y}$ denote the prompts and responses, and $r(\cdot,\cdot)$ is the reward function.

Our Perspective: RLHF Should Be Made Open-Ended

Definition 1 (Open-Ended RLHF) *We define* evolving alignment *as the open-ended joint optimization on the prompt and response policy for alignment w.r.t the joint reference policy:*

$$\max_{\phi,\theta} \mathbb{E}_{\mathbf{x}\sim\pi_\phi(\cdot),\ \mathbf{y}\sim\pi_\theta(\cdot|\mathbf{x})}\Big[r(\mathbf{x},\mathbf{y})\Big] - \beta\cdot\mathbb{D}_{KL}\Big[\pi_{\phi,\theta}(\mathbf{x},\mathbf{y})\,\|\,\pi_{ref}(\mathbf{x},\mathbf{y})\Big], \quad (7)$$

where $\pi_{\phi,\theta}(\mathbf{x},\mathbf{y}) := \pi_\phi(\mathbf{x})\cdot\pi_\theta(\mathbf{y}\mid\mathbf{x})$ *and* $\pi_{ref}(\mathbf{x},\mathbf{y}) := p_{ref}(\mathbf{x})\cdot\pi_{SFT}(\mathbf{y}\mid\mathbf{x})$[a].

[a] This generalizes classical RLHF (Eq. 1). One may extend the above and rewrite coefficients to be:

$$\max_{\phi,\theta} \mathbb{E}_{\mathbf{x}\sim\pi_\phi(\cdot)}\Big[\mathbb{E}_{\mathbf{y}\sim\pi_\theta(\cdot|\mathbf{x})}\big[r(\mathbf{x},\mathbf{y})\big] - \beta_1\mathbb{D}_{\text{KL}}\big[\pi_\theta(\mathbf{y}|\mathbf{x})\,\|\,\pi_{\text{SFT}}(\mathbf{y}|\mathbf{x})\big]\Big] - \beta_2\,\mathbb{D}_{\text{KL}}\Big[\pi_\phi(\mathbf{x})\,\|\,p_{\text{ref}}(\mathbf{x})\Big]. \quad (8)$$

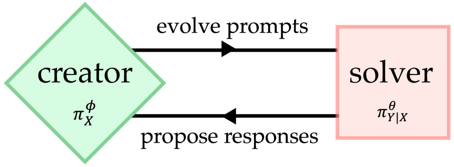
However, directly optimizing this can be *intractable* or *unstable*... 🤔

See also: Open-Endedness is Essential for Artificial Superhuman Intelligence

Our Method: Open-Ended RLHF via Creator-Solver Games

- How? **Optimization** by **Asymmetric Games**

$$\max_{\boldsymbol{\phi},\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x}\sim\pi_\phi(\cdot)}\Big[\underbrace{\mathbb{E}_{\mathbf{y}\sim\pi_\theta(\cdot|\mathbf{x})}\big[r(\mathbf{x},\mathbf{y}) \big] - \beta_1 \cdot \mathbb{D}_{\mathrm{KL}}\left[\pi_\theta(\mathbf{y} \mid \mathbf{x}) \,\|\, \pi_{\mathrm{SFT}}(\mathbf{y} \mid \mathbf{x})\right]}_{\text{solver}} \Big] - \beta_2 \cdot \underbrace{\mathbb{D}_{\mathrm{KL}}\big[\pi_\phi(\mathbf{x}) \,\|\, p_{\mathrm{ref}}(\mathbf{x}) \big]}_{\text{creator}}$$



- What? The **Regret** of the Solver's Policy

$$\mathrm{Regret}(\pi_\phi, \pi_{\boldsymbol{\theta}}) = \mathbb{E}_{\mathbf{x}\sim\pi_\phi(\cdot)}\Big[\mathbb{E}_{\mathbf{y}\sim\pi_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x})}\big[r(\mathbf{x},\mathbf{y}) \big] - \mathbb{E}_{\mathbf{y}\sim\pi^\star_{\mathrm{KL}}(\mathbf{y}|\mathbf{x})}\big[r(\mathbf{x},\mathbf{y}) \big]\Big]$$

- Why? **The Minimax Regret Strategy** at the Nash Equilibrium

$$\pi^\star_{\mathcal{Y}|\mathcal{X}} \in \arg\min_{\pi_{\mathcal{Y}|\mathcal{X}}} \max_{\pi_{\mathcal{X}}} \mathbb{E}_{\mathbf{x}\sim\pi_{\mathcal{X}}}\Big[\mathrm{Regret}(\mathbf{x}, \pi_{\mathcal{Y}|\mathcal{X}}) \Big]$$

However, w/o access to the true π^*, we must **approximate** this regret... 🤔

Our Method: Open-Ended RLHF via Creator-Solver Games

- How to approximate the **regret**? Simply use the **stochastic policy...**

Sample N times from the policy,
then choosing the *reward gap* between *the best* and *the baseline*.

$$|\widehat{\text{Regret}}(\mathbf{x}, \pi_{\boldsymbol{\theta}})| \leftarrow \text{info}_{\boldsymbol{\theta}}(\mathbf{x}) := r(\mathbf{x}, \mathbf{y}_+) - r(\mathbf{x}, \mathbf{y}_{\text{baseline}}),$$

$$\mathbf{y}_+ := \arg\max_{\mathbf{y}_i} r(\mathbf{x}, \mathbf{y}),$$
$$\mathbf{y}_{\text{baseline}} := \arg\min_{\mathbf{y}_i} r(\mathbf{x}, \mathbf{y}) \text{ or } \mathbf{y}_{\text{baseline}} := \text{avg}_{\mathbf{y}_i} \; r(\mathbf{x}, \mathbf{y})$$

- Other intuitive interpretations of **`eva`**

Learning potential.
`eva` picks the prompts that are learnable but not learned yet.

Auto-curricula for the solver player.
The optimal strategy of the creator is to create prompts just beyond solvers' current capability .

Worst-case guarantee.
The minimax objective incentivizes the solver to perform well in all cases.

Auto-curricula inherent to contrastive learning.
`eva` prioritizes prompts with lower contrastive loss by design, thus accelerating learning.

See also: PAIRED: A New Multi-Agent Framework for Adversarial Environment Generation

The eva Algorithm

Algorithm 1 **eva: Evolving Alignment *via* Asymmetric Self-Play**

Input: initial policy $\pi_{\boldsymbol{\theta}_0}$, initial prompt set $\mathcal{X}_0$

1: **for iteration** $t = 1, 2, \ldots$ **do**

▽ /* creator step */

2: *estimate informativeness*: $\mathcal{X}_{t-1} \leftarrow \{(\mathbf{x}_i, \texttt{info}(\mathbf{x}_i)) \mid \mathbf{x}_i \in \mathcal{X}_{t-1}\}$

sample subset: $\mathcal{X}_{t-1}^{\texttt{info}} \leftarrow \texttt{sample}(\mathcal{X}_{t-1})$

self-evolve prompts: $\mathcal{X}_t \leftarrow \texttt{evolve}(\mathcal{X}_{t-1}^{\texttt{info}})$

▽ /* solver step */

3: *self-generate responses*: $\forall\, \boldsymbol{x}_i \in \mathcal{X}_t$, generate $\{\boldsymbol{y}_i^{(j)}\} \sim \pi_{\boldsymbol{\theta}_{t-1}}(\cdot \mid \boldsymbol{x}_i)$

annotate rewards: $\mathcal{X}_t' \leftarrow \mathcal{X}_t \cup \{(\boldsymbol{y}_i^{(j)}, r_i^{(j)})\}$

preference optimization: $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{X}_t'}(\boldsymbol{\theta})$

4: **end for**

5: **return** final solver policy $\pi_{\boldsymbol{\theta}_T}$

Fig 6. **eva** requires only **a creator module addition** to make current RLHF pipeline **open-ended**.

The Creator Step: Estimate, Sample then Evolve

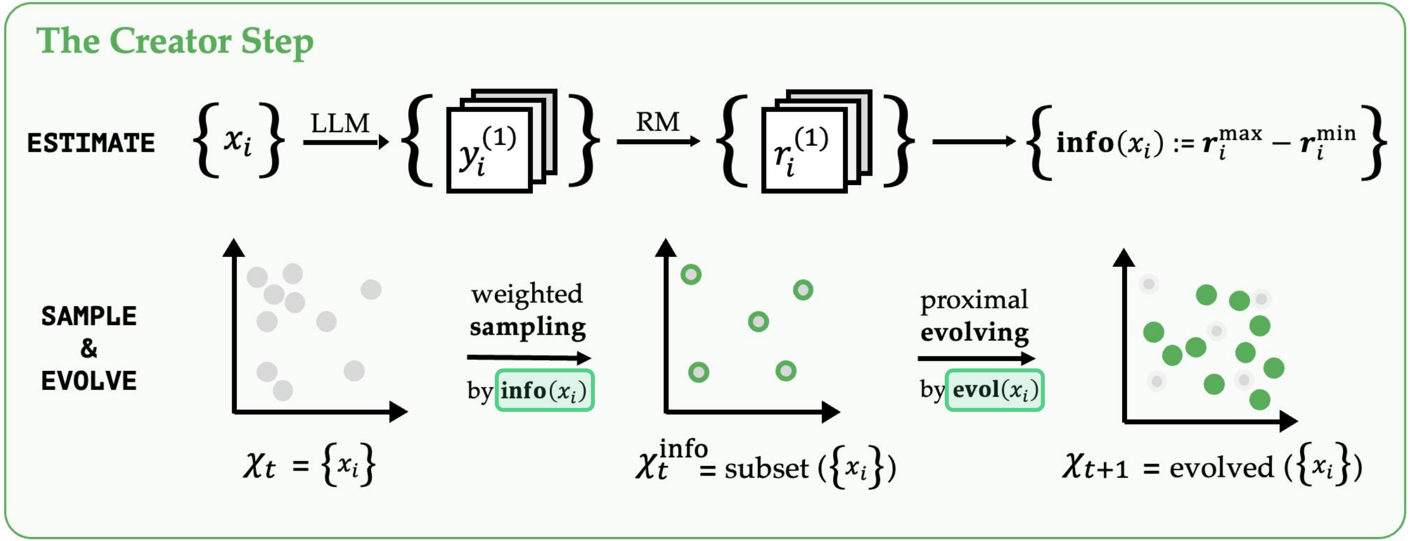


Fig 7. **eva** currently uses the **estimate, sample then evolve** procedure for the creator. Here, `info(·)` is the reward gap, and `evol(·)` can be any prompt creation method.

Example Evolving Method: `evol(·)`

We use **EvolInstruct** (Can et al., 2023) for **in-depth evolving** and **in-breadth-evolving**.

Initial prompt ↓

If a man smokes 1000 cigarettes a day, why is he getting healthier?

Evolved #1 ↓ (*in-depth* evolving)

Elaborate on the seemingly paradoxical situation where an individual consumes 1000 cigarettes daily yet exhibits signs of improving health, delineating the factors that could underlie such an unexpected outcome.

Evolved #2 ↓ (*in-breadth* evolving)

Discuss the conundrum of a person drinking a gallon of caffeinated coffee every hour but displaying unusually deep and restful sleep patterns, exploring possible explanations for this unusual phenomenon.

```
MUTATION_TEMPLATES = {
    # ------------------------------------
    # >>>>>>>> In-depth evolving <<<<<<<<
    # ------------------------------------
    "CONSTRAINTS": prompt.format(
        "Add one more constraints into '#The Given Prompt#'"
    ),
    "DEEPENING": prompt.format(
        "If #The Given Prompt# contains inquiries about certain issues,
        the depth and breadth of the inquiry can be increased."
    ),
    "CONCRETIZING": prompt.format(
        "Please replace general concepts with more specific concepts."
    ),
    "INCREASED_REASONING_STEPS": prompt.format(
        "If #The Given Prompt# can be solved with just a few simple
        thinking processes, you can rewrite it to explicitly request
        multiple-step reasoning."
    ),

    # ------------------------------------
    # >>>>>>>> In-breadth evolving <<<<<<<<
    # ------------------------------------
    "BREADTH": prompt.format(
        "By inspiration from #The Given Prompt#, create a new prompt.
        This new prompt should belong the the same domain as it, but be
        even more rare. The length and complexity should be similar. The
        #Created Prompt# must be reasonable and must be understood and
        responded by humans."
    )
```

Results: Remarkable Gains on Hard Benchmarks*!

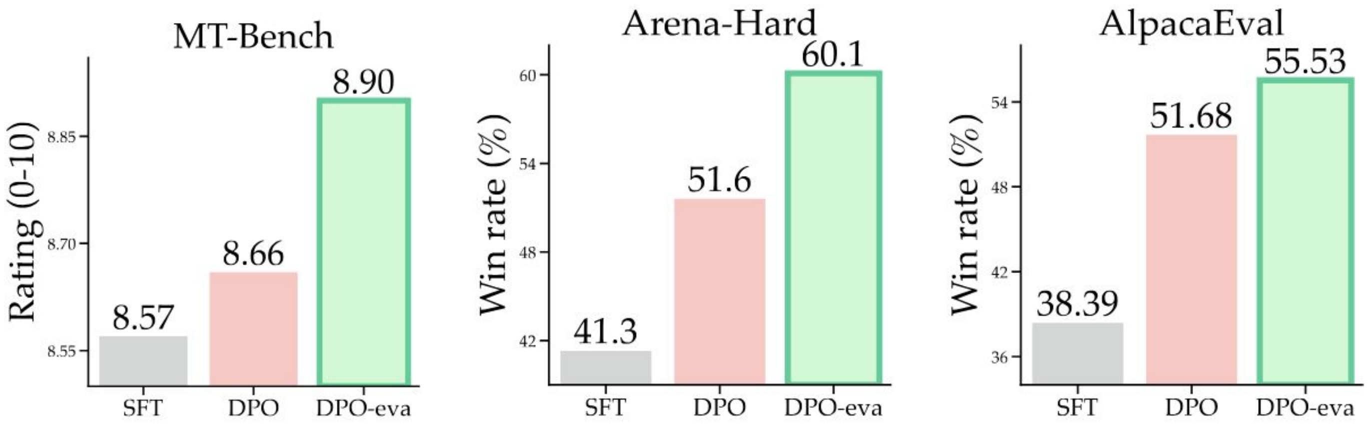


Figure. **`eva`** achieves concrete performance gain especially on **hard benchmarks**, without relying on any additional human prompts.

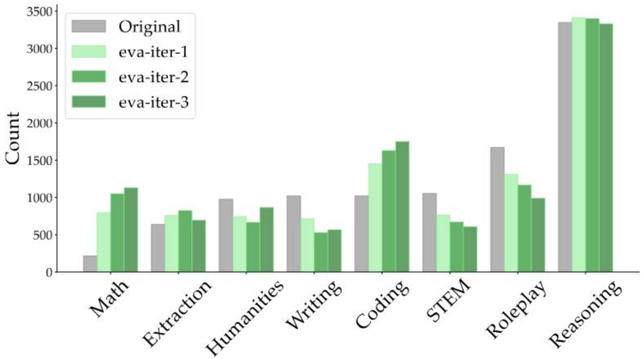
Results: Remarkable Gains on Hard Benchmarks*!

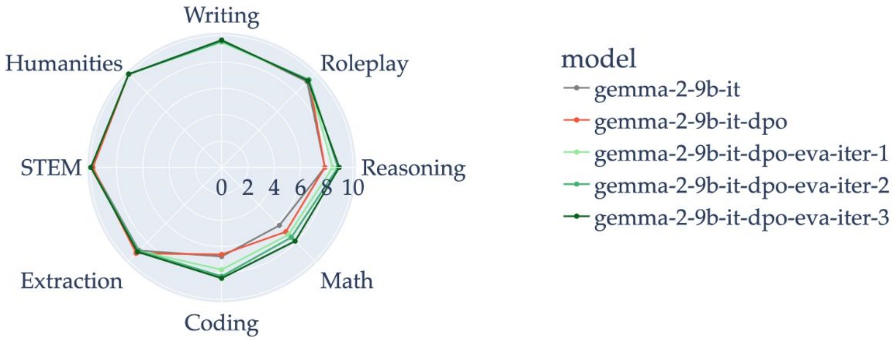
Model Family (→)	GEMMA-2-9B-IT					
Benchmark (→)	**Arena-Hard**	**MT-Bench**			**AlpacaEval 2.0**	
Method (↓) / **Metric** (→)	**WR** (%)	**avg. score**	1^{st} turn	2^{nd} turn	**LC-WR** (%)	**WR** (%)
θ_0: SFT	41.3	8.57	8.81	8.32	47.11	38.39
$\theta_{0\to1}$: DPO	51.6	8.66	9.01	8.32	55.01	51.68
$\theta_{1\to\tilde{1}}$: + **eva**	**60.1** (+8.5)	**8.90**	**9.04**	**8.75** (+0.43)	55.35	55.53
$\theta_{1\to2}$: + new human prompts	59.8	8.64	8.88	8.39	**55.74**	**56.15**
$\theta_{0\to1}$: SPPO	55.7	8.62	9.03	8.21	51.58	42.17
$\theta_{1\to\tilde{1}}$: + **eva**	**58.9** (+3.2)	**8.78**	**9.11**	**8.45** (+0.24)	**51.86**	**43.04**
$\theta_{1\to2}$: + new human prompts	57.7	8.64	8.90	8.39	51.78	42.98
$\theta_{0\to1}$: SimPO	52.3	8.69	9.03	8.35	54.29	52.05
$\theta_{1\to\tilde{1}}$: + **eva**	**60.7** (+8.4)	**8.92**	**9.08**	**8.77** (+0.42)	**55.85**	**55.92**
$\theta_{1\to2}$: + new human prompts	54.6	8.76	9.00	8.52	54.40	55.72
$\theta_{0\to1}$: ORPO	54.8	8.67	9.04	8.30	52.17	49.50
$\theta_{1\to\tilde{1}}$: + **eva**	**60.3** (+5.5)	**8.89**	**9.07**	**8.71** (+0.41)	**54.39**	50.88
$\theta_{1\to2}$: + new human prompts	57.2	8.74	9.01	8.47	54.00	**51.21**

Table 1: **Main results.** Our **eva** achieves notable alignment gains and can surpass human prompts on major benchmarks across a variety of representative direct preference optimization algorithms.

* All experiments are conducted with external open-source frameworks and models on HuggingFace.
We use 10K prompts from UltraFeedback for training, and use ArmoRM-8B as the default reward model.

Additional Results – **eva** creates meaningful curriculum.





Prompt Set (↓) / **Metric** (→)	**Complexity** (1-5)	**Quality** (1-5)
UltraFeedback (seed)	2.90	3.18
UltraFeedback-**eva**-Iter-1	3.84	3.59
UltraFeedback-**eva**-Iter-2	3.92	3.63
UltraFeedback-**eva**-Iter-3	**3.98**	**3.73**

Ablation #1 – **eva**'s minimax design outperforms alternatives.

Metric	info(x)	Related Interpretations
$A^\star_{\text{min}}$: worst-case optimal advantage	$\lvert \min_{\mathbf{y}} r(\mathbf{x},\mathbf{y}) - \max_{\mathbf{y}'} r(\mathbf{x},\mathbf{y}')\rvert$	minimax regret (Savage, 1951)
$A^\star_{\text{avg}}$: average optimal advantage	$\lvert \frac{1}{N}\sum_{\mathbf{y}} r(\mathbf{x},\mathbf{y}) - \max_{\mathbf{y}'} r(\mathbf{x},\mathbf{y}')\rvert$	Bayesian regret (Banos, 1968)
$A^\star_{\text{dts}}$: dueling optimal advantage	$\lvert \max_{\mathbf{y}\neq\mathbf{y}^\star} r(\mathbf{x},\mathbf{y}) - \max_{\mathbf{y}'} r(\mathbf{x},\mathbf{y}')\rvert$	dueling regret (Wu and Liu, 2016)

Table 2: The reward-advantage-based metrics that serve as the informativeness proxies for prompts.

Model Family (→)	GEMMA-2-9B-IT					
Benchmark (→)	Arena-Hard	MT-Bench			AlpacaEval 2.0	
Method (↓) / Metric (→)	WR (%)	avg. score	1st turn	2nd turn	LC-WR (%)	WR (%)
$\boldsymbol{\theta}_{0\to1}$: DPO	51.6	8.66	9.01	8.32	55.01	51.68
$\boldsymbol{\theta}_{1\to\tilde{1}}$: + **eva** (`uniform`)	57.5	8.71	9.02	8.40	53.43	53.98
$\boldsymbol{\theta}_{1\to\tilde{1}}$: + **eva** (`var`($\boldsymbol{r}$))	54.8	8.66	9.13	8.20	54.58	52.55
$\boldsymbol{\theta}_{1\to\tilde{1}}$: + **eva** (`avg`($\boldsymbol{r}$))	58.5	8.76	9.13	8.40	55.01	55.47
$\boldsymbol{\theta}_{1\to\tilde{1}}$: + **eva** (1/`avg`($\boldsymbol{r}$))	56.7	8.79	9.13	8.45	55.04	54.97
$\boldsymbol{\theta}_{1\to\tilde{1}}$: + **eva** ($1/A^\star_{\text{min}}$)	52.3	8.64	8.96	8.31	53.84	52.92
$\boldsymbol{\theta}_{1\to\tilde{1}}$: + **eva** ($A^\star_{\text{avg}}$) (our variant)	60.0	8.85	9.08	8.61	**56.01**	**56.46**
$\boldsymbol{\theta}_{1\to\tilde{1}}$: + **eva** ($A^\star_{\text{dts}}$) (our variant)	60.0	8.86	**9.18**	8.52	55.96	56.09
$\boldsymbol{\theta}_{1\to\tilde{1}}$: + **eva** ($A^\star_{\text{min}}$) (our default)	**60.1** (+8.5)	**8.90**	9.04	**8.75** (+0.43)	55.35	55.53

Table 3: **Choice of informativeness metric.** Our informativeness metric by *advantage* achieves the best performances, comparing with others as the weight to sample prompts to evolve by the creator.

Ablation #2 – **eva**'s design of evolving is meaningful.

Benchmark (→)	**Arena-Hard**	**MT-Bench**			**AlpacaEval 2.0**	
Method (↓) / Metric (→)	**WR (%)**	**avg. score**	1^{st} turn	2^{nd} turn	**LC-WR** (%)	**WR** (%)
$\boldsymbol{\theta}_{0\to1}$: DPO	51.6	8.66	9.01	8.32	55.01	51.68
$\boldsymbol{\theta}_{1\to\tilde{1}}$: `[no evolve]-greedy`	56.1	8.68	8.98	8.38	54.11	53.66
$\boldsymbol{\theta}_{1\to\tilde{1}}$: `[no evolve]-sample`	55.3	8.69	9.00	8.38	54.22	54.16
$\boldsymbol{\theta}_{1\to\tilde{1}}$: + **`eva`**`-greedy` (our variant)	59.5	8.72	9.06	8.36	54.52	55.22
$\boldsymbol{\theta}_{1\to\tilde{1}}$: + **`eva`**`-sample` (our default)	**60.1**	**8.90**	9.04	**8.75**	**55.35**	**55.53**

Table 4: **Effect of evolving.** The blue are those training w/ only the informative subset and w/o evolving); we denote `-sample` for the default weighted sampling procedure in Algo 1, while using `-greedy` for the variant from the classical active data selection procedure (*cf.*, a recent work (Muldrew et al., 2024) and a pre-LLM work (Kawaguchi and Lu, 2020)), which selects data by a high-to-low ranking via the metric greedily. We show evolving brings a remarkable alignment gain (the red v.s. the blue); and as we evolve, sampling is more robust than being greedy (*cf.*, Russo et al. (2018)).

Ablation #3 – **eva** scales with better reward models.

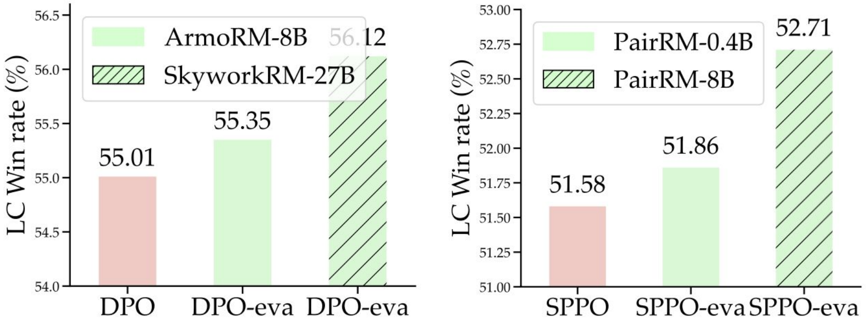


Figure. **eva** scales with better reward models.

Ablation #4 – **eva** is robust in continual training.

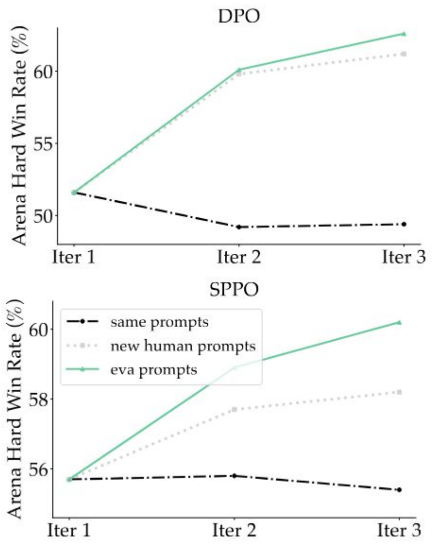


Figure 5: **Continual training. eva** stays robust w/ more iterations in incremental training.

Model Family (→)	GEMMA-2-9B-IT	
Benchmark (→)	Arena-Hard	
Method (↓) / Metric (→)	WR (%)	avg. len
θ_0: SFT	41.3	544
$\theta_{0\to1}$: DPO (10k)	51.6	651
$\theta_{1\to2}$: DPO (10k)	59.8	718
$\theta_{2\to3}$: DPO (10k)	61.2	802
$\theta_{1\to\tilde{1}}$: + **eva** (10k)	60.1	733
$\theta_{\tilde{1}\to\tilde{2}}$: + **eva** (10k)	62.0	787
$\theta_{\tilde{2}\to\tilde{3}}$: + **eva** (10k)	62.2	774

Model Family (→)	GEMMA-2-9B-IT	
Benchmark (→)	Arena-Hard	
Method (↓) / Metric (→)	WR (%)	avg. len
θ_0: SFT	41.3	544
$\theta_{0\to1}$: DPO (20k)	53.2	625
$\theta_{1\to2}$: DPO (20k)	47.0	601
$\theta_{2\to3}$: DPO (20k)	46.8	564
$\theta_{1\to\tilde{1}}$: + **eva** (20k)	59.5	826
$\theta_{\tilde{1}\to\tilde{2}}$: + **eva** (20k)	60.0	817
$\theta_{\tilde{2}\to\tilde{3}}$: + **eva** (20k)	61.4	791

Takeaways

`eva` is a new, simple framework for aligning language models via a creator-solver game.

RLHF can be made **open ended**:

- **self-evolving joint data distributions** (with synthesized prompts) bring significant gains.
- **reward advantage** acts as an effective metric for prompt selection.

[work-in-progress]

Self-Play to Reason: Decompose + Search is All You Need

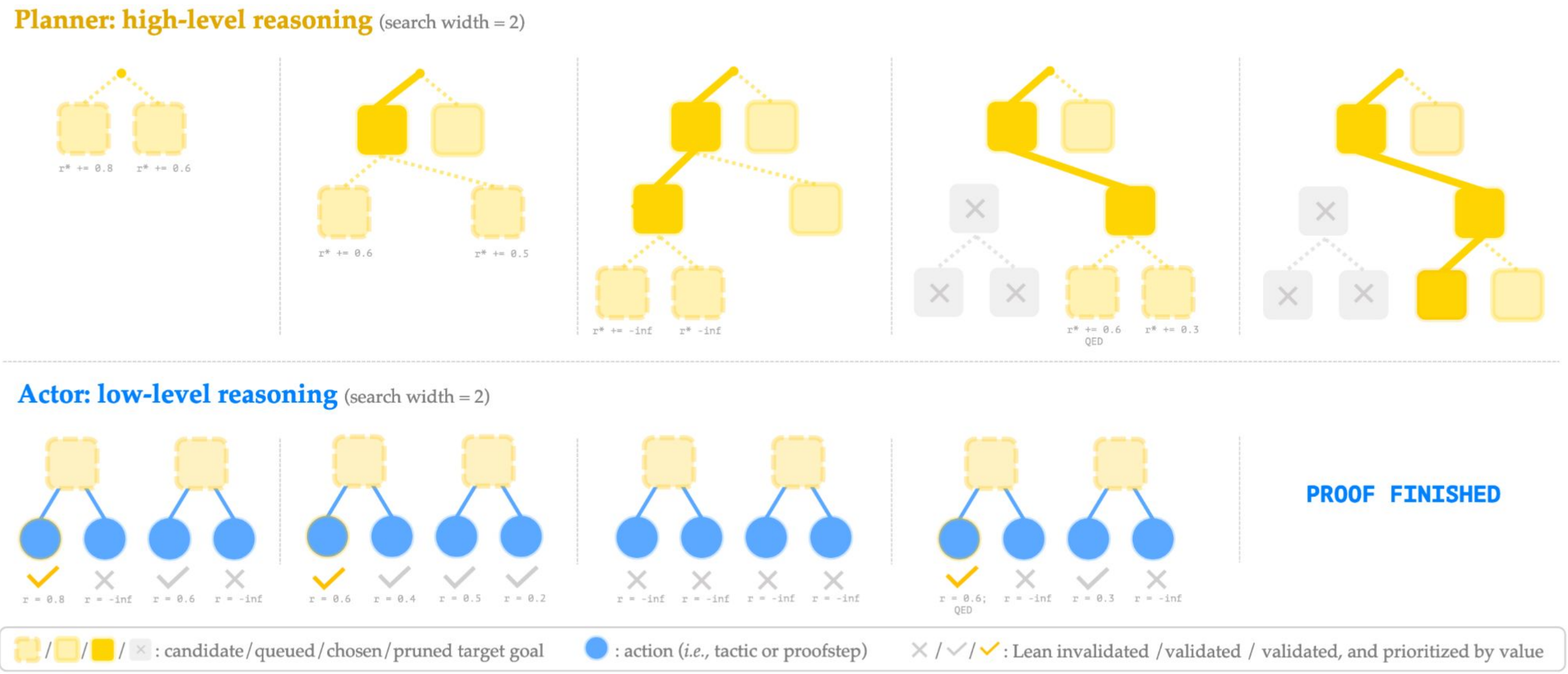
"Better than state-of-the-art and 3x faster for neural theorem proving."

Gratitude to every wonderful co-author of this project (link):
Jiacheng Chen, Jonathan Light, Yifei Wang, Jiankai Sun, Mac Schwager, Guohao Li, Philip Torr, Yuxin Chen, Kaiyu Yang, Yisong Yue, Ziniu Hu.

THE UNIVERSITY OF CHICAGO

TL; DR

We unify **decomposing** and **search** for **better and faster reasoning**.



Preliminaries

```
theorem (p q: Prop): p v q  q v p := by        -- goal s0: (p q: Prop) p v q  q v p
    intro h                                    -- goal s1: (p q: Prop)(h: p v q)  q v p
    cases h with                               -- goal s2: (p q: Prop)(hp: p)  q v p
     inl hp => apply Or.inr; exact hp          -- goal s3: (p q: Prop)(hq: q)  q v p
     inr hq => apply Or.inl; exact hq          -- goal s4: None
```

Neural theorem proving. A neural network parameterized by $\boldsymbol{\theta}$ can act as a policy that samples single tactic $\mathbf{y}_{t+1} \sim \pi_{\boldsymbol{\theta}}(\cdot \mid \mathbf{s}_t)$ at step t. The objective is to find the optimal trajectory which leads to `solved` for each statement $\mathbf{q}$, that is to find a sequence of tactics $\mathbf{y}_1, \ldots, \mathbf{y}_T$ such that:

$$\mathbf{s}_0 \xrightarrow{\mathbf{y}_1} \mathbf{s}_1 \xrightarrow{\mathbf{y}_2} \mathbf{s}_2 \xrightarrow{\mathbf{y}_3} \ldots \xrightarrow{\mathbf{y}_T} \mathbf{s}_T.$$

Classical training method.

```
> Input:   {$current_goal s}
> Output:  {$proofstep y*}
```

Intuition for Flat Search v.s. Hierarchical Search

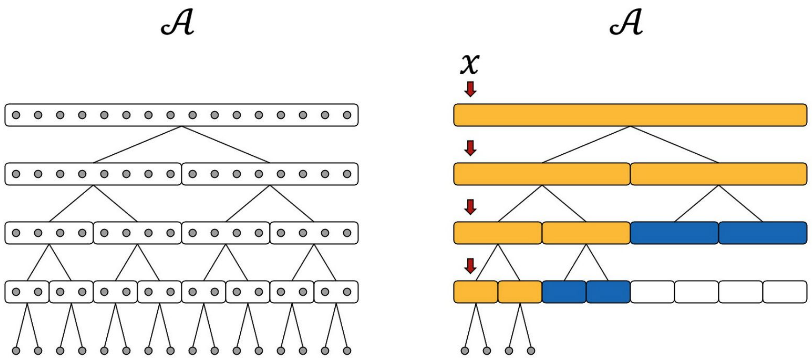


Figure 1. Hierarchical decomposition for the flat action space; the yellow nodes are further explored [Reference].

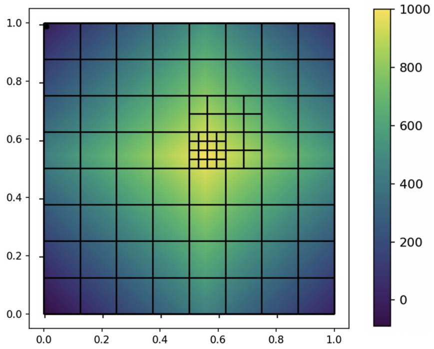


Figure 2. Partitioning over the action space [Reference].

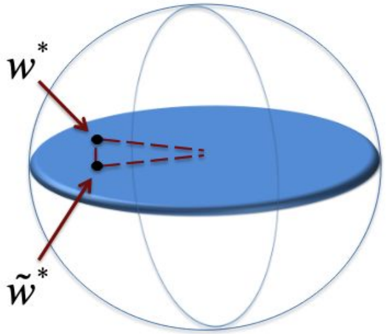


Figure 3. Focused exploration in subspaces [Reference].

Method: (Offline SFT Stage) Goal-Driven Co-Training

$$\mathcal{L}_{\text{co}}(\boldsymbol{\theta}) = -\frac{1}{N} \underbrace{\sum_{(\mathbf{s},\mathbf{y}^\star,\mathbf{s}^\star)\sim\mathcal{D}^{\text{train}}}}_{\text{triplet set}} \Big[\underbrace{\log p_{\boldsymbol{\theta}}(\mathbf{s}^\star \mid \mathbf{s})}_{\text{goal planner}} + \underbrace{\log p_{\boldsymbol{\theta}}(\mathbf{y}^\star \mid \mathbf{s}, \mathbf{s}^\star)}_{\text{goal-driven actor}} \Big]$$

Let's think in an information-theoretic way: $\mathbf{s}_{t+1}$ **acts as an information bottleneck** [Shwartz-Ziv and Tishby, 2017], by *abstracting* different possible proofsteps or sequences of proofsteps $\mathbf{y}_t$ into a single, more compact representation. Consider a simplified example below:

```
-- goal s_t   =  3 * (2 + 1) = 9
-- goal s_t+1 =  9 = 9
```

There exist multiple different proofsteps to reach $\mathbf{s}_{t+1}$ from $\mathbf{s}_t$, for instance:

- `ring` – algebraic normalization.
- `norm_num` – direct numeric evaluation.
- `simp; rfl` – simplification followed by reflexivity.
- `calc ··· (omitted)` – step-by-step calculation.

Method: (Online Search Stage) Goal-Driven Hierarchical Search

Algorithm 1 **RiR** – *A Unified Reasoning Mechanism with Decomposing and Search*

Input: problem statement $\boldsymbol{q}$, a language model w/ parameter $\boldsymbol{\theta}$

$\overline{\texttt{tree}} \leftarrow \overline{\texttt{Tree}}(\boldsymbol{\theta}, \boldsymbol{q})$
repeat
  $\boldsymbol{s}_l^\star \leftarrow \overline{\texttt{tree}}\texttt{.policy()}$ ▷ /* planner */
  $\underline{\texttt{tree}} \leftarrow \underline{\texttt{Tree}}(\boldsymbol{\theta}, \boldsymbol{s}_l^\star)$
  repeat
    $\boldsymbol{y}_{t_l} \leftarrow \underline{\texttt{tree}}\texttt{.policy()}$ ▷ /* goal-driven actor */
  until STOP_LOW
  $\{\overline{\texttt{tree}}, \underline{\texttt{tree}}\}\texttt{.update()}$ ▷ /* joint update */
until STOP_HIGH
return $\underline{\texttt{tree}}\texttt{.solution}$

- (Classical) **Flat planning**: we have a policy $\pi_f : \mathcal{S} \rightarrow \mathcal{A}$ that maps states to actions.
- (RiR) **Hierarchical planning**: we have:
  - A *high-level planner* policy $\pi_h : \mathcal{S} \rightarrow \tilde{\mathcal{S}}$, that maps goals to target goals.
  - A *low-level actor* policy $\pi_l : \mathcal{S} \times \tilde{\mathcal{S}} \rightarrow \mathcal{A}$, that maps goals and target goals to actions.

Results: Robust *Performance* Gains

Search Method (→)	**Best-First Search**	
Dataset (→)	**miniF2F-test**[2]	**LeanDojo-test**
Method (↓) / Model (→)	BYT5-0.3B	BYT5-0.3B
Reprover	34.43%	50.16%
RiR	**36.89%**	**53.73%**

Table 1: **Performance with BFS.** *Pass@1* rate on LeanDojo and miniF2F.

Search Method (→)	**Monte-Carlo Tree Search**	
Dataset (→)	**miniF2F-test**	**LeanDojo-test**
Method (↓) / Model (→)	BYT5-0.3B	BYT5-0.3B
Reprover	36.51%	50.24%
RiR	**37.83%**	**53.92%**

Table 2: **Performance with MCTS.** *Pass@1* rate on LeanDojo and miniF2F.

Results: Remarkable *Efficiency* Gains

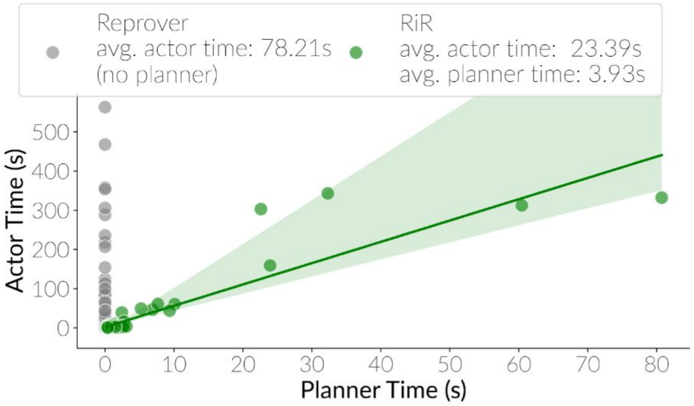


Figure 2: **Efficiency**. The scatter plot for actor and planner time spent for proved theorems on miniF2F. RiR significantly reduces the actor time via the goal guidance from the planner.

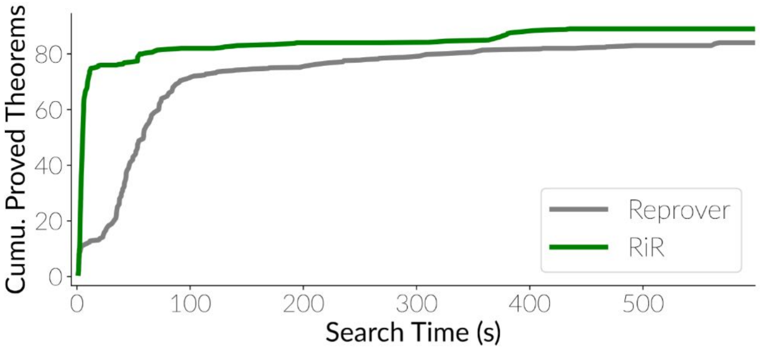


Figure 3: **Efficiency**. The CDF plot for search time spent for proved theorems on miniF2F Benchmark. RiR is significantly faster (nearly **3x**) than the existing state-of-the-art baseline.

Example Proofs

Example 3: Proof Found by RiR

```
Theorem:
  File Path: Mathlib/Order/SuccPred/Basic.lean
  Full Name: exists_succ_iterate_or

Status: Status.PROVED

Proof:
  obtain h | h := le_total a b
  exacts [Or.inl (IsSuccArchimedean.exists_succ_iterate_of_le h),
  Or.inr (IsSuccArchimedean.exists_succ_iterate_of_le h)]

Search Statistics:
  Planner Time: 15.921687303110957
  Actor Time: 44.464585242792964
  Environment Time: 8.429574175737798
  Total Time: 68.86368872597814
  Total Nodes: 377
  Searched Nodes: 3
```

Example 3: Failure by Reprover (w/o retrieval)

```
Theorem:
  File Path: Mathlib/Order/SuccPred/Basic.lean
  Full Name: exists_succ_iterate_or

Status: Status.OPEN

Proof: None

Search Statistics:
  Actor Time: 519.0408471203409
  Environment Time: 86.30267171841115
  Total Time: 605.4483464460354
  Total Nodes: 2819
  Searched Nodes: 95
```

Takeaways

`RiR` is a hierarchical framework for complex reasoning, unifying **decomposing** and **search**, and is **significantly faster** than classical stepwise reasoning, with **robust performance gains**.

The performance and efficiency gains come from:

- **Offline co-training for SFT**.
- **Online bi-level search.**

p.s., There are many different ways for decomposing!

What Next?

rigorous theories + more practical applications

Q & A