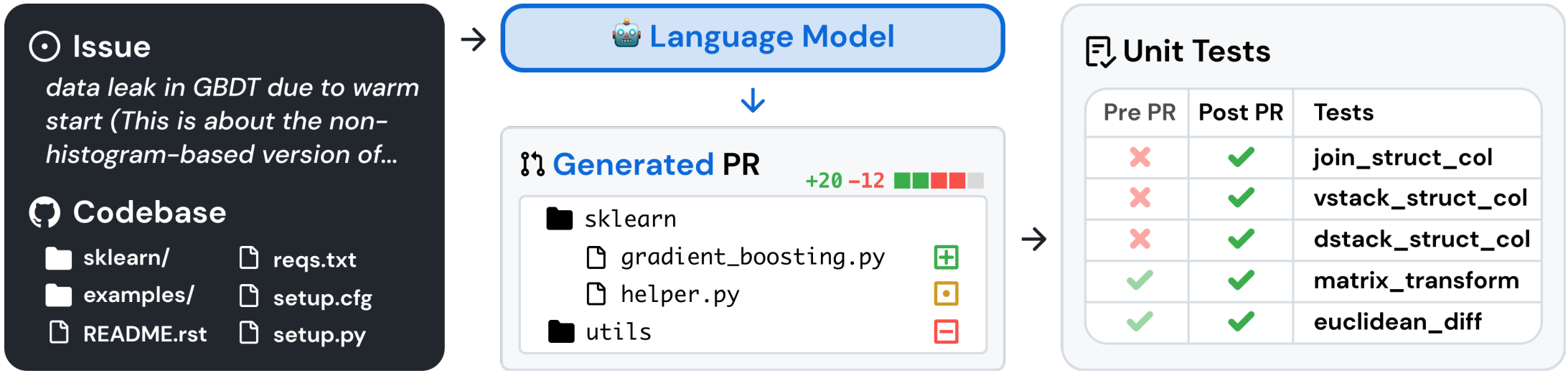
Satori-SWE: Evolutionary Test-Time Scaling for Sample-Efficient Software Engineering

Guangtao Zeng

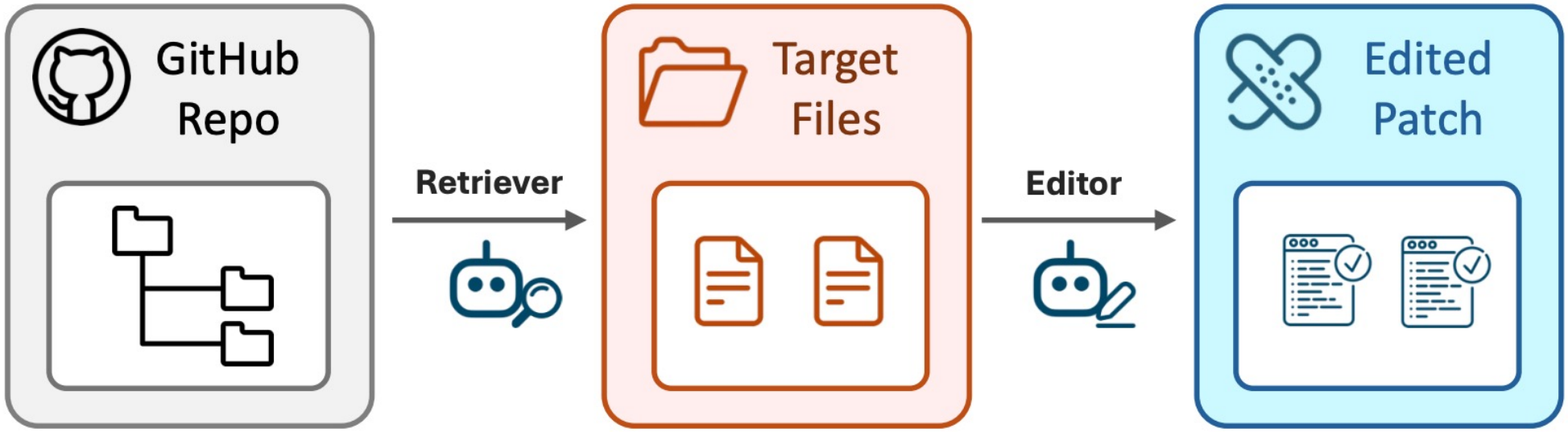
5th July

Overview for Software Engineering Task



- https://github.com/SWE-bench/SWE-bench

Overview for Software Engineering Pipeline



Importance of Software Engineering Task

Weeks per task

Hours per task

Over-Reliance on Proprietary Models

Large gap between Proprietary Models and Open Source small model

Proprietary Models

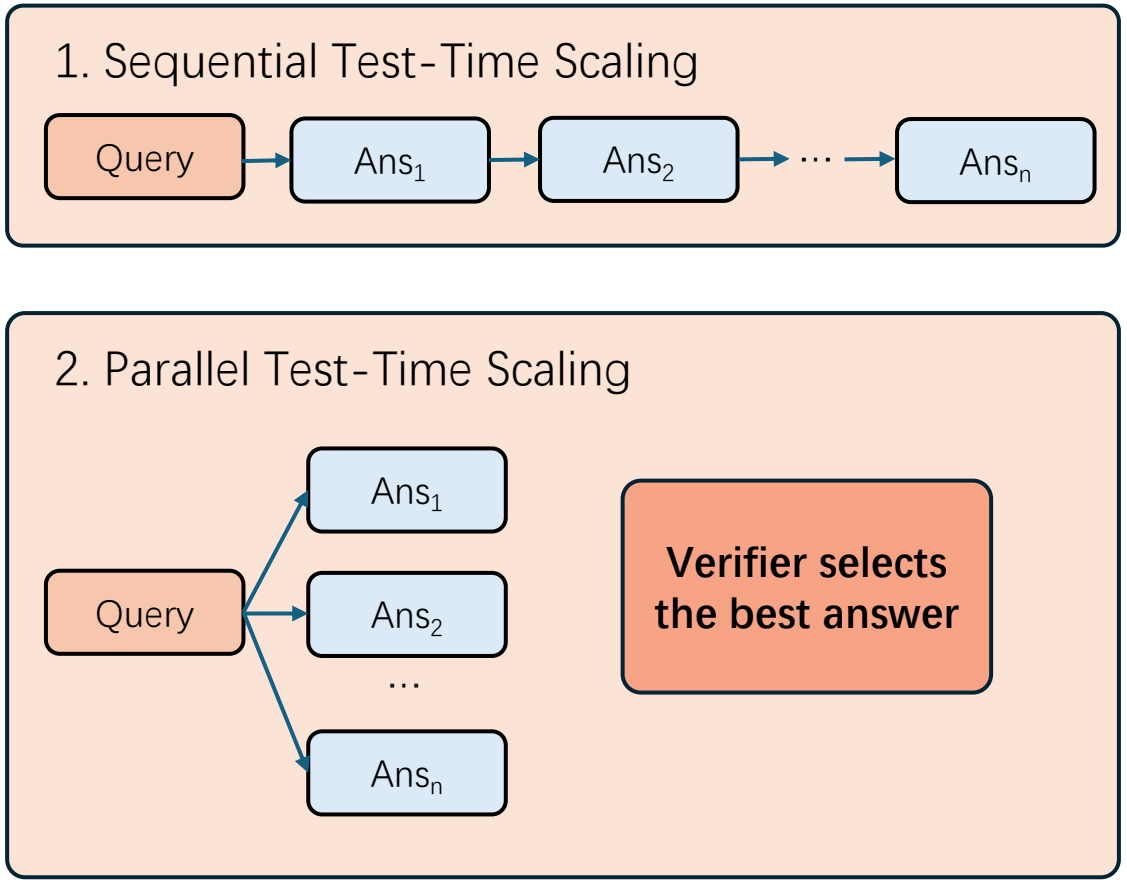
Model	% Resolved	Org	Date	Logs	Trajs	Site
Moatless Tools + Claude 4 Sonnet	70.80	-	2025-06-11	✓	✓	
Refact.ai Agent	70.40		2025-05-15	✓	✓	
OpenHands + Claude 4 Sonnet	70.40		2025-05-24	✓	✓	
Augment Agent v1	70.40		2025-06-10	✓	✓	
SWE-agent + Claude 4 Sonnet	66.60		2025-05-22	✓	✓	
OpenHands	65.80		2025-04-15	✓	✓	

Open Source Small Models

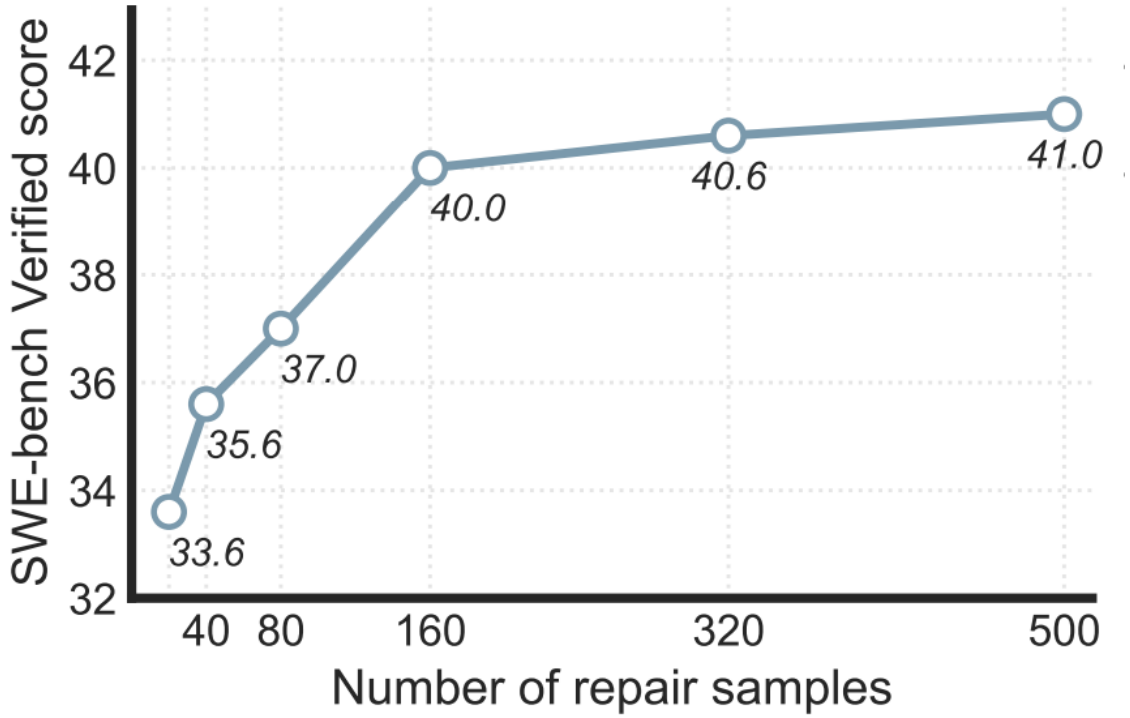
Model	% Resolved	Org	Date	Logs	Trajs	Site
Skywork-SWE-32B + TTS(Bo8)	47.00		2025-06-16	✓	✓	
OpenHands + DevStral Small 2505	46.80		2025-05-20	✓	✓	
PatchPilot + Co-PatcheR	46.00		2025-05-28	✓	✓	
SWE-agent + SWE-agent-LM-32B	40.20		2025-05-11	✓	✓	
Skywork-SWE-32B	38.00		2025-06-16	✓	✓	
SWE-Fixer (Qwen2.5-7b retriever + Qwen2.5-72b editor)	32.80		2025-03-06	✓	✓	

- https://github.com/SWE-bench/SWE-bench

Test-Time Scaling Boosts Model Performance on SWE-bench



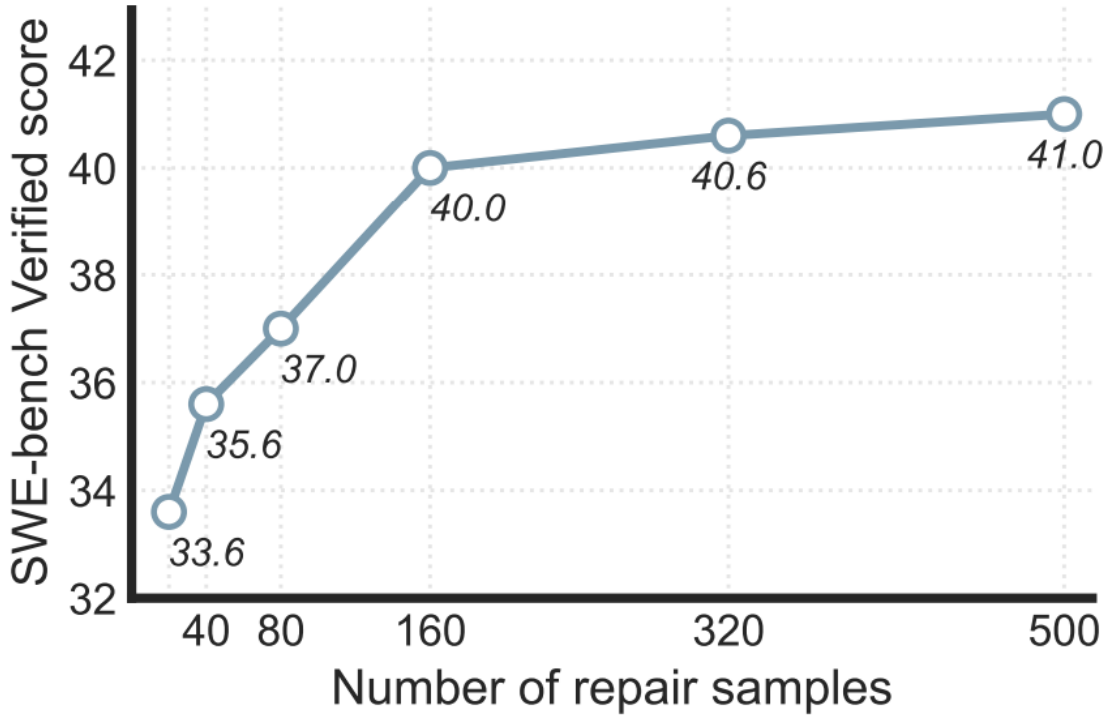
Test-Time Scaling Boosts Model Performance on SWE-bench



- Score improves from 33.6 to 41.0 as repair samples increase.

- SWE-RL: Advancing LLM Reasoning via Reinforcement Learning on Open Software Evolution

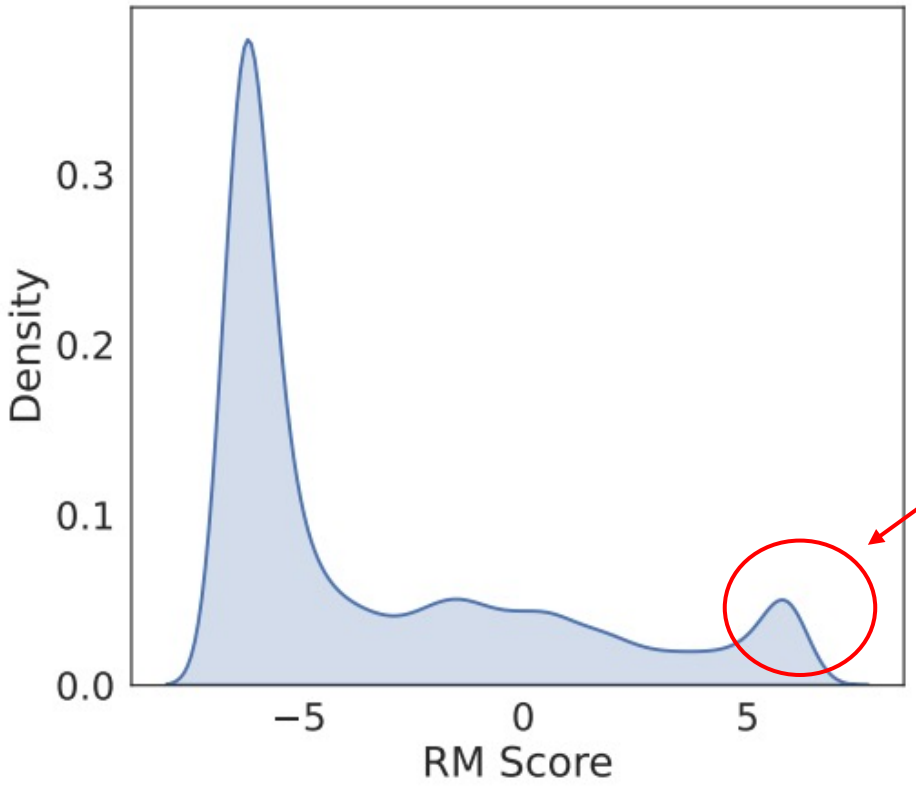
Test-Time Scaling Boosts Model Performance on SWE-bench



- Score improves from 33.6 to 41.0 as repair samples increase.
- **Sample-inefficient!**
  - **Unit test running in docker. (few mins per issue)**
  - **Long context in inference.**

- SWE-RL: Advancing LLM Reasoning via Reinforcement Learning on Open Software Evolution

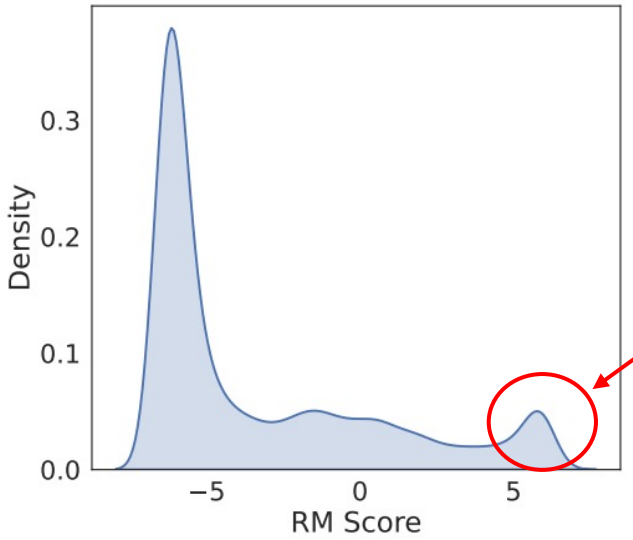
Why is test-time scaling sample-inefficient in SWE task?



high-scoring outputs

in the long tail

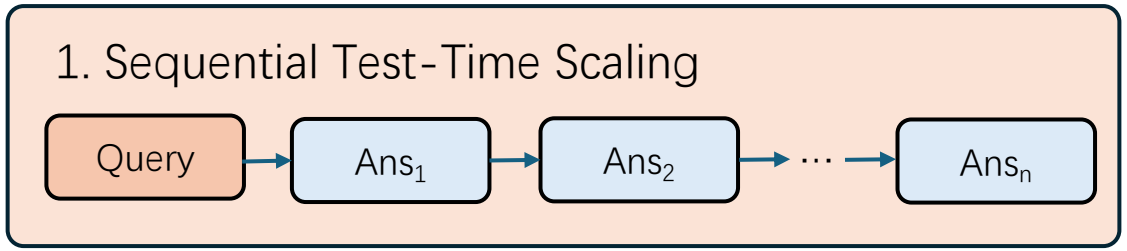
Why is test-time scaling sample-inefficient in SWE task?

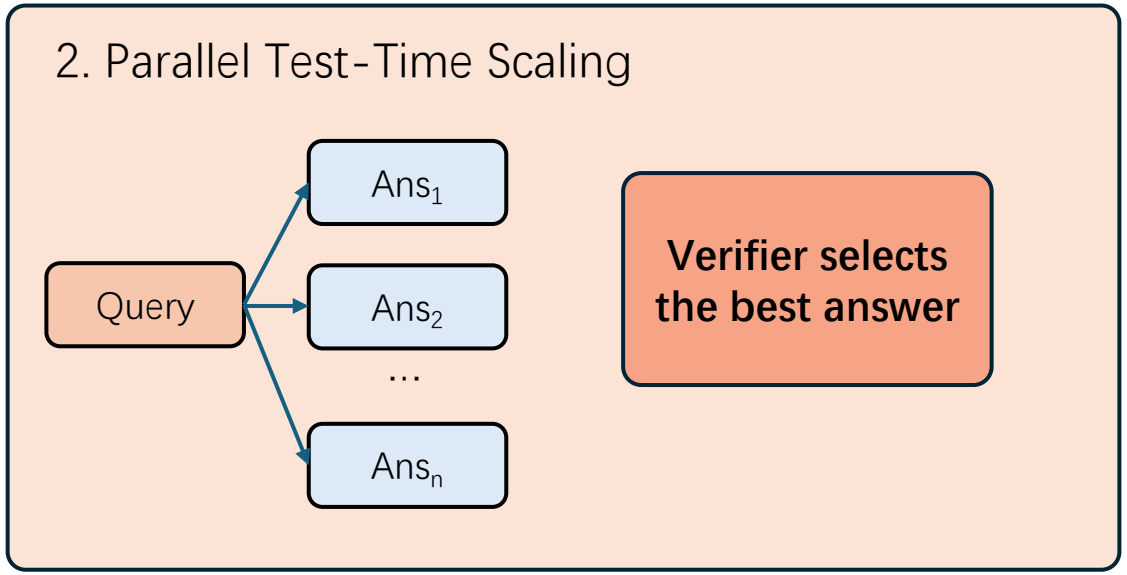


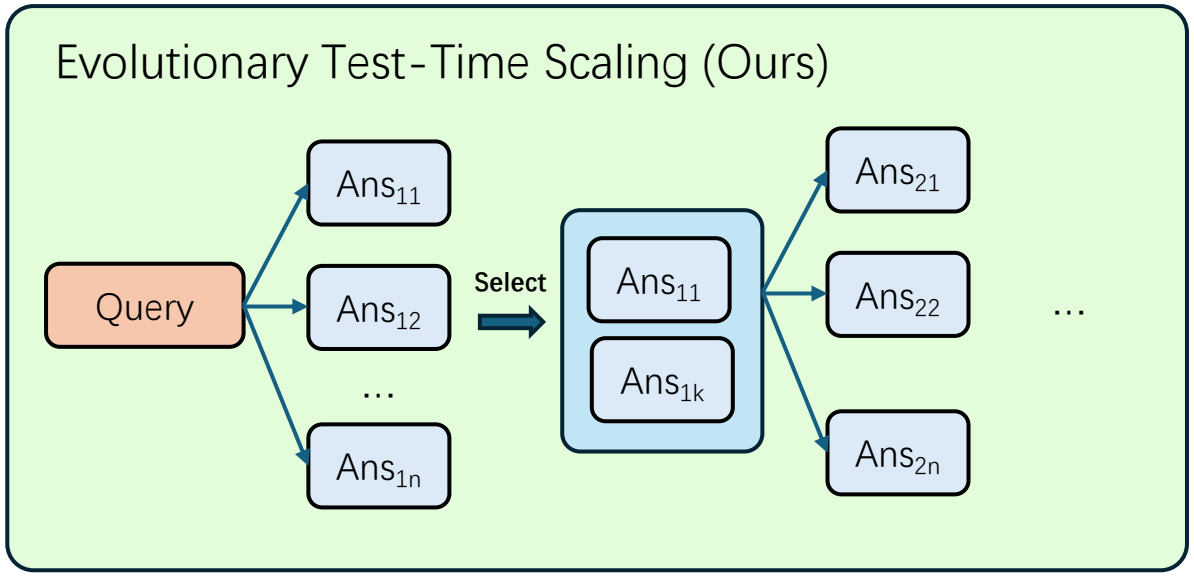
high-scoring outputs in the long tail

Can we teach model to change the **distribution of its generation toward the high score?**

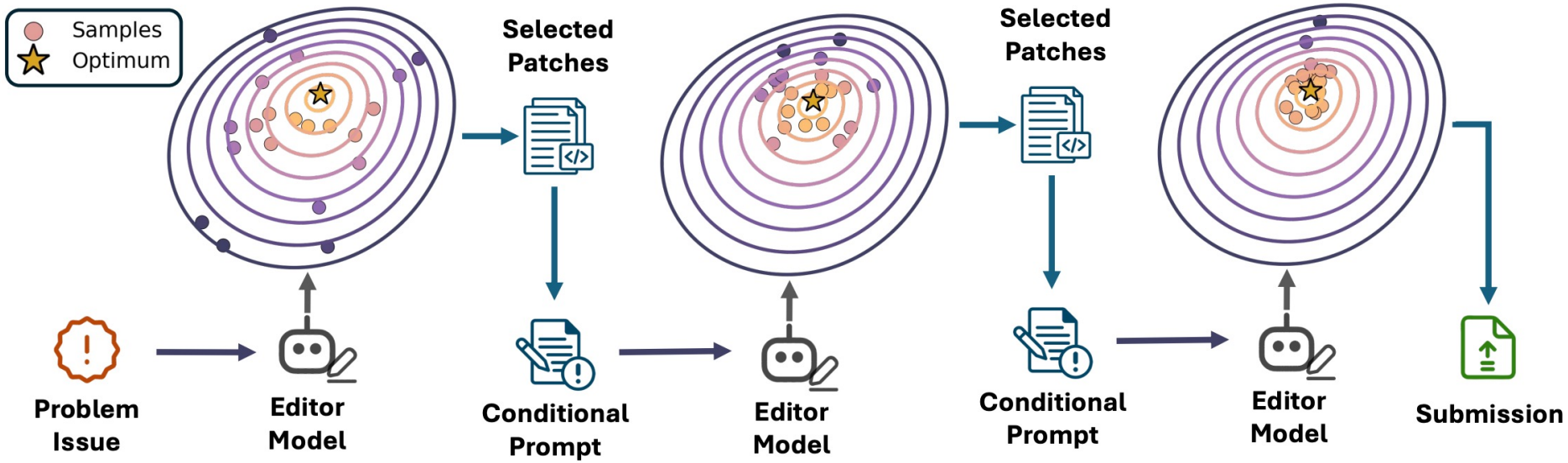
Compared to existed Test-Time Scaling methods







Evolutionary Test-Time Scaling Pipeline



Classical SFT

- SFT **objective function**

$$\max_{\pi_{\mathrm{SFT}}} \mathbb{E}_{x\sim\mathcal{D},\, y^*_{\mathrm{SFT}}\sim\mu(\cdot|x,C(x))} \left[\log \pi_{\mathrm{SFT}}(y^*_{\mathrm{SFT}} \mid x, C(x))\right].$$

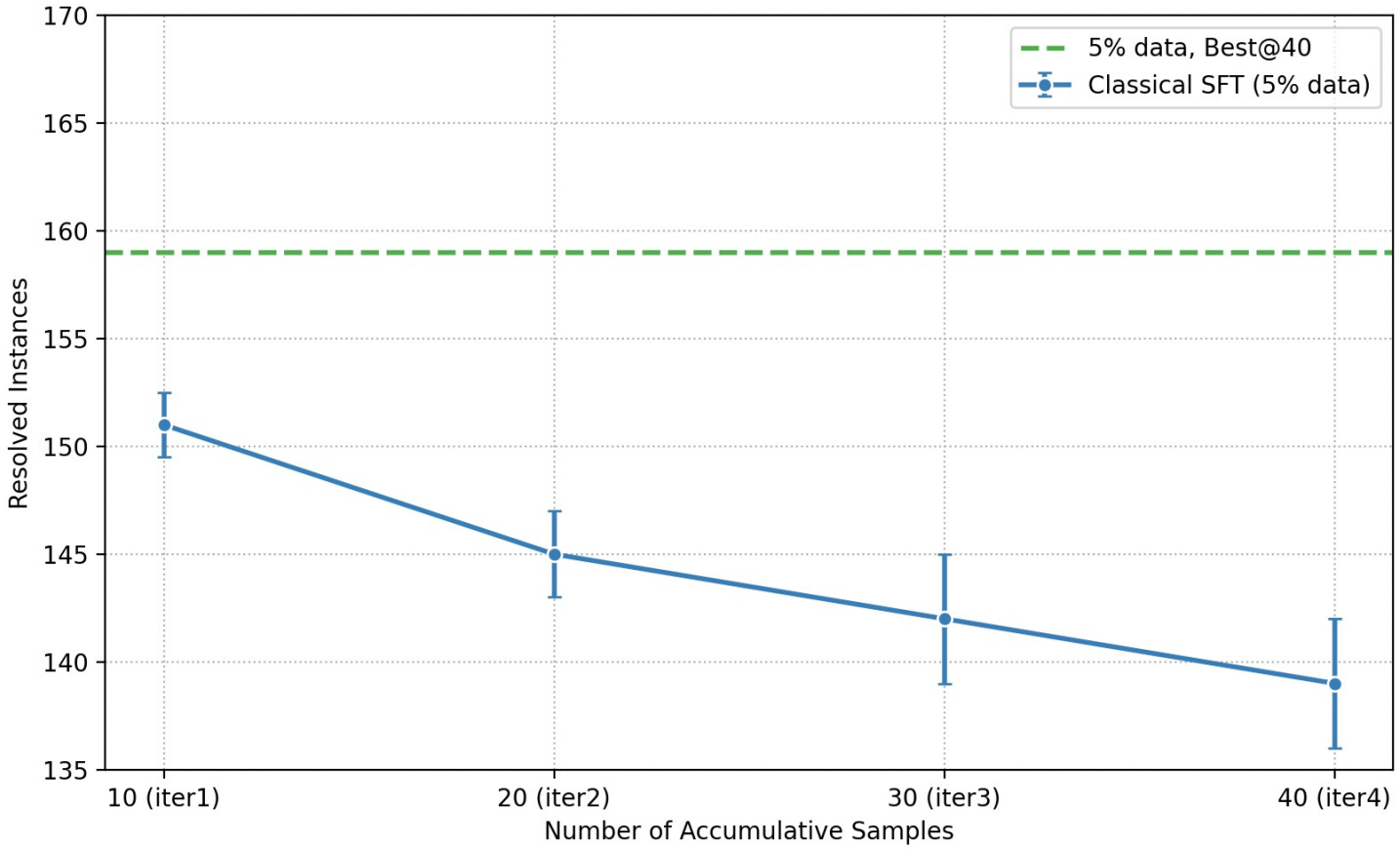
- SFT data

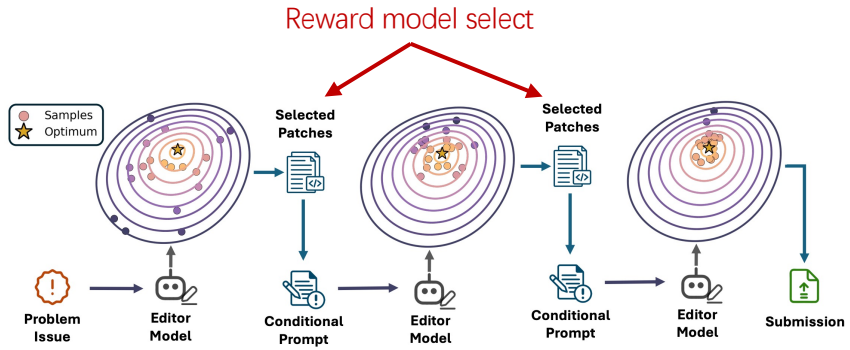
Problem Statement

<Thinking>

Oracle Patch Answer

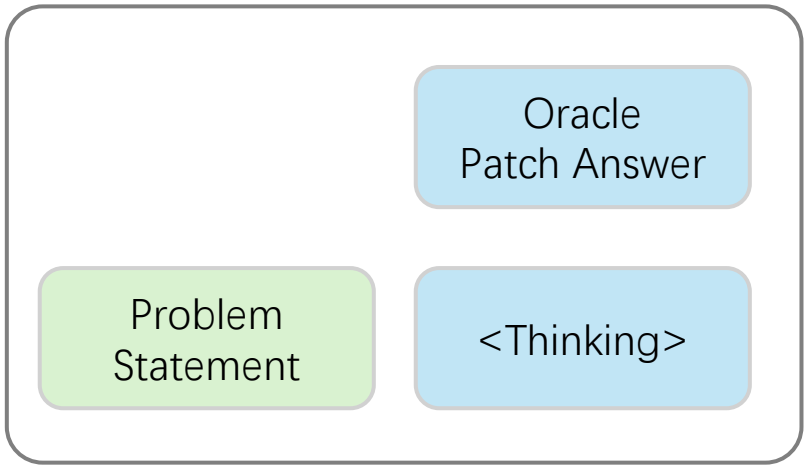
Model with only Classical SFT can not have the ability to evolve.



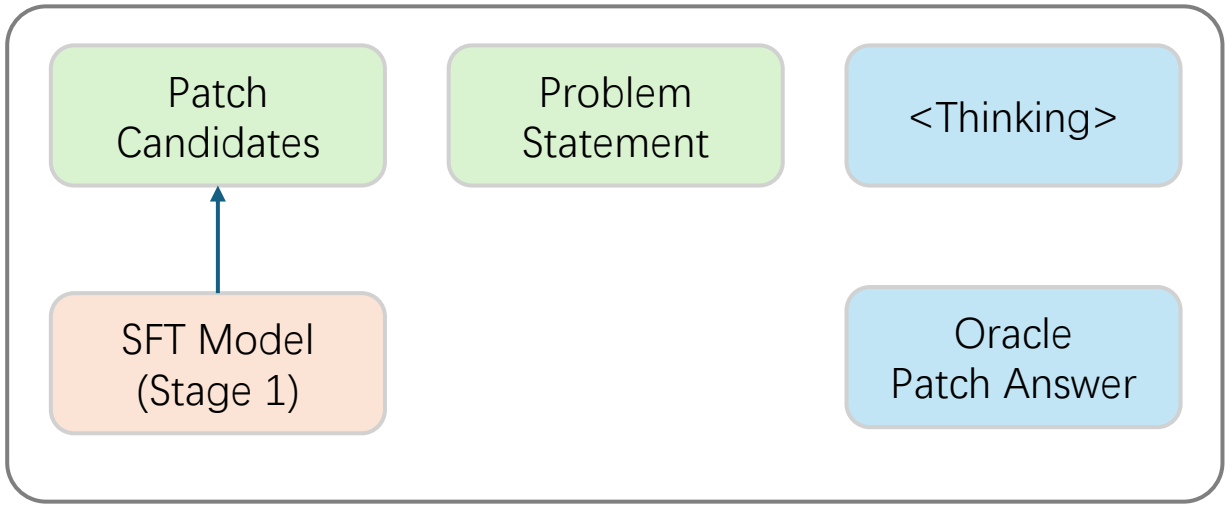


Mutation-Based SFT Enable Models Evolve

- **Mutation SFT** （mutation data +classical data）

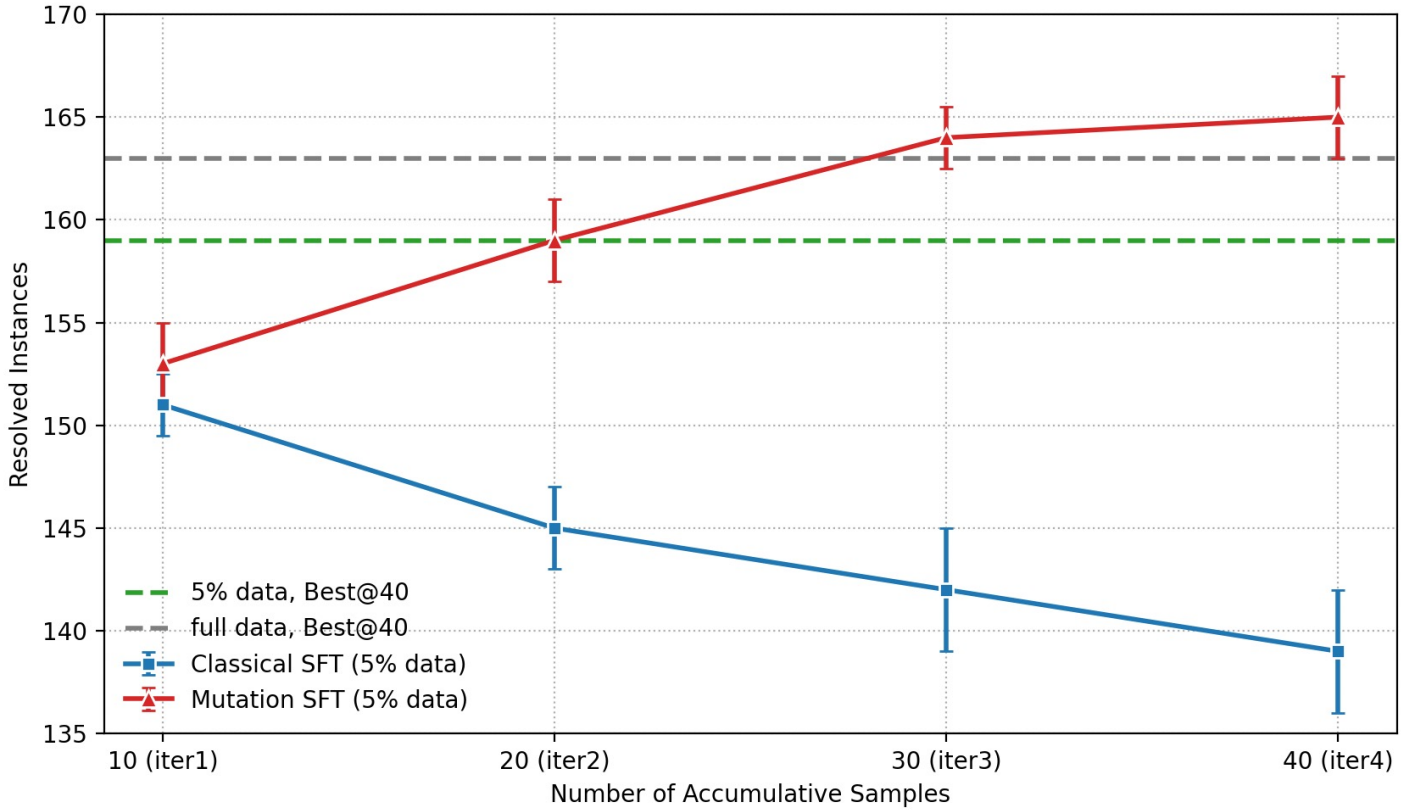


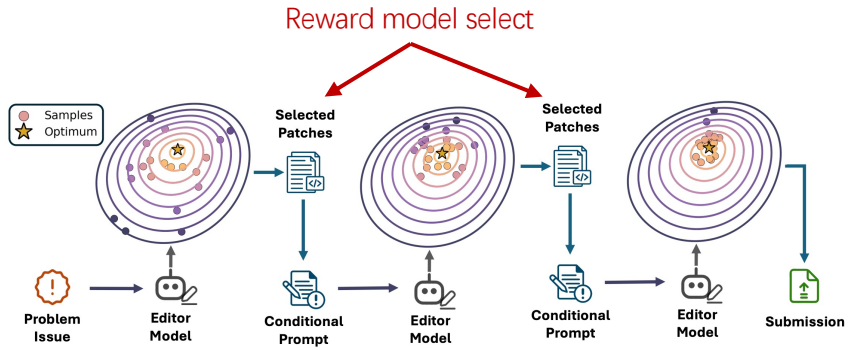
Classical data



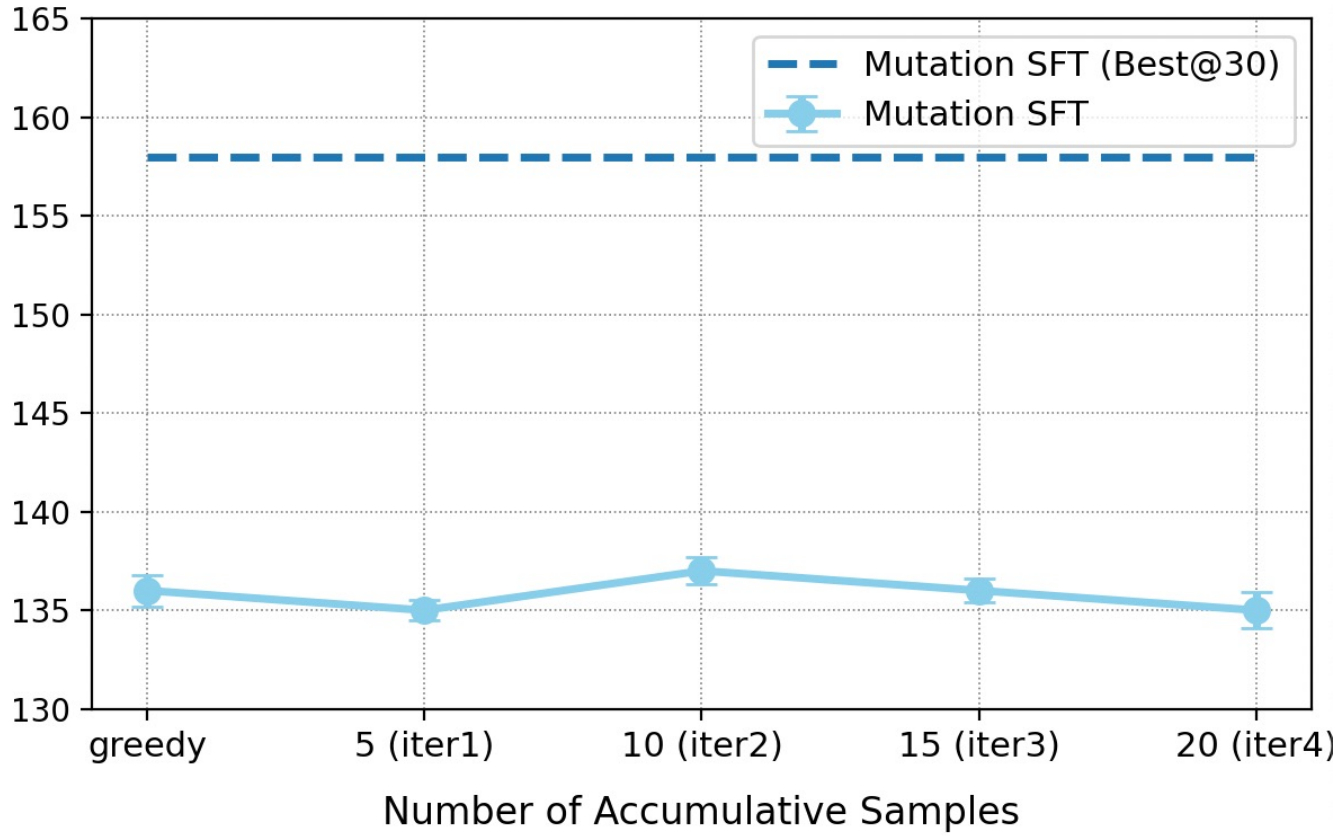
Mutation data

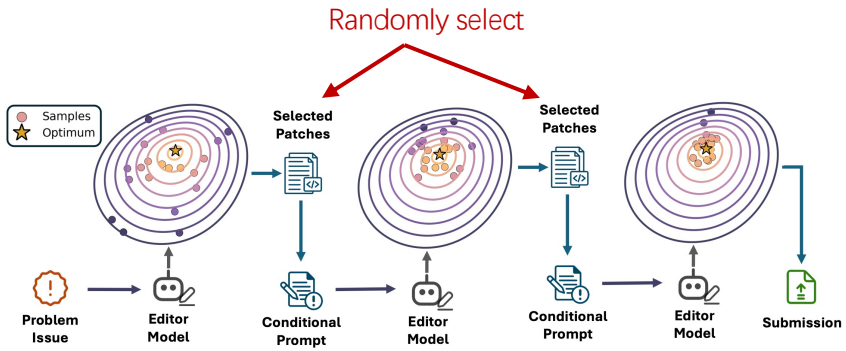
Mutation SFT helps LLMs iteratively evolve



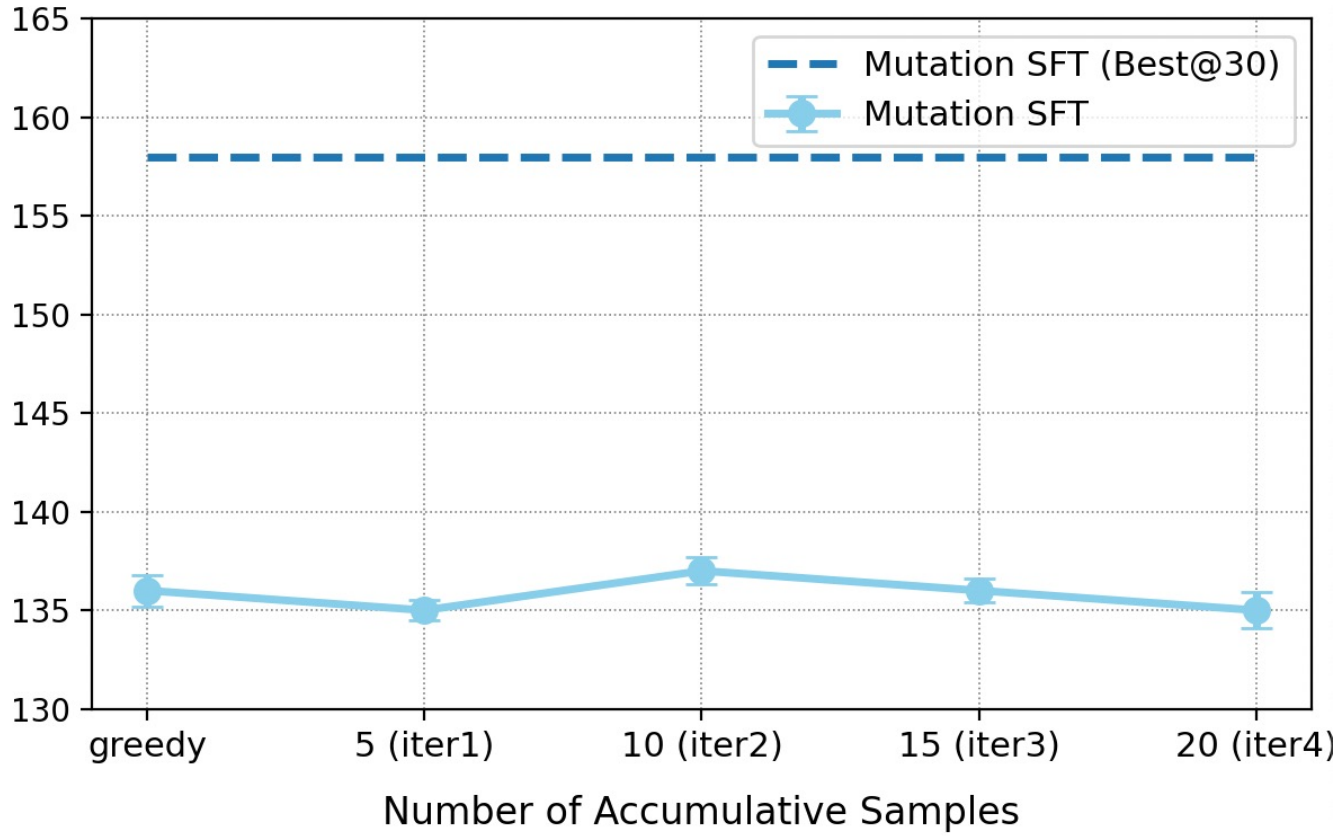


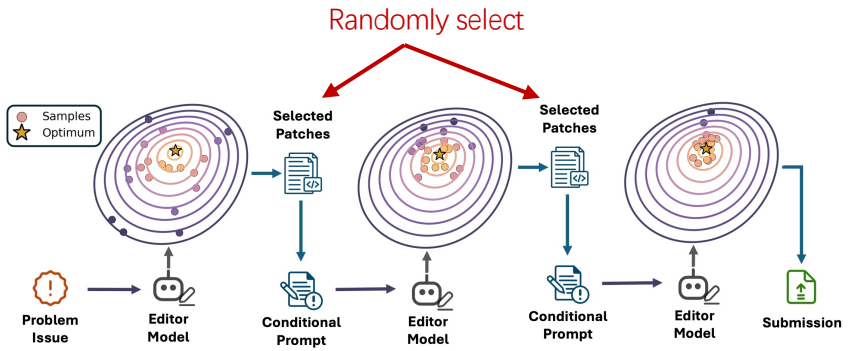
However, SFT-only Model fails to self-improve without reward model





However, SFT-only Model fails to self-improve without reward model





We then introduce an RL approach that teaches model to **self-evolve without scoring or filtering.**

Learning to Self-evolve via Large-scale RL

- Naïve RL objective is:

$$\max_{\pi} \mathbb{E}_{y^t \sim \pi(\cdot|x, C(x), \mathcal{E}^{t-1})} \left[\sum_{t=0}^{T} r_t \right], \quad \text{where} \quad r_t = \begin{cases} R(x, y^t), & t = T \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Learning to Self-evolve via Large-scale RL

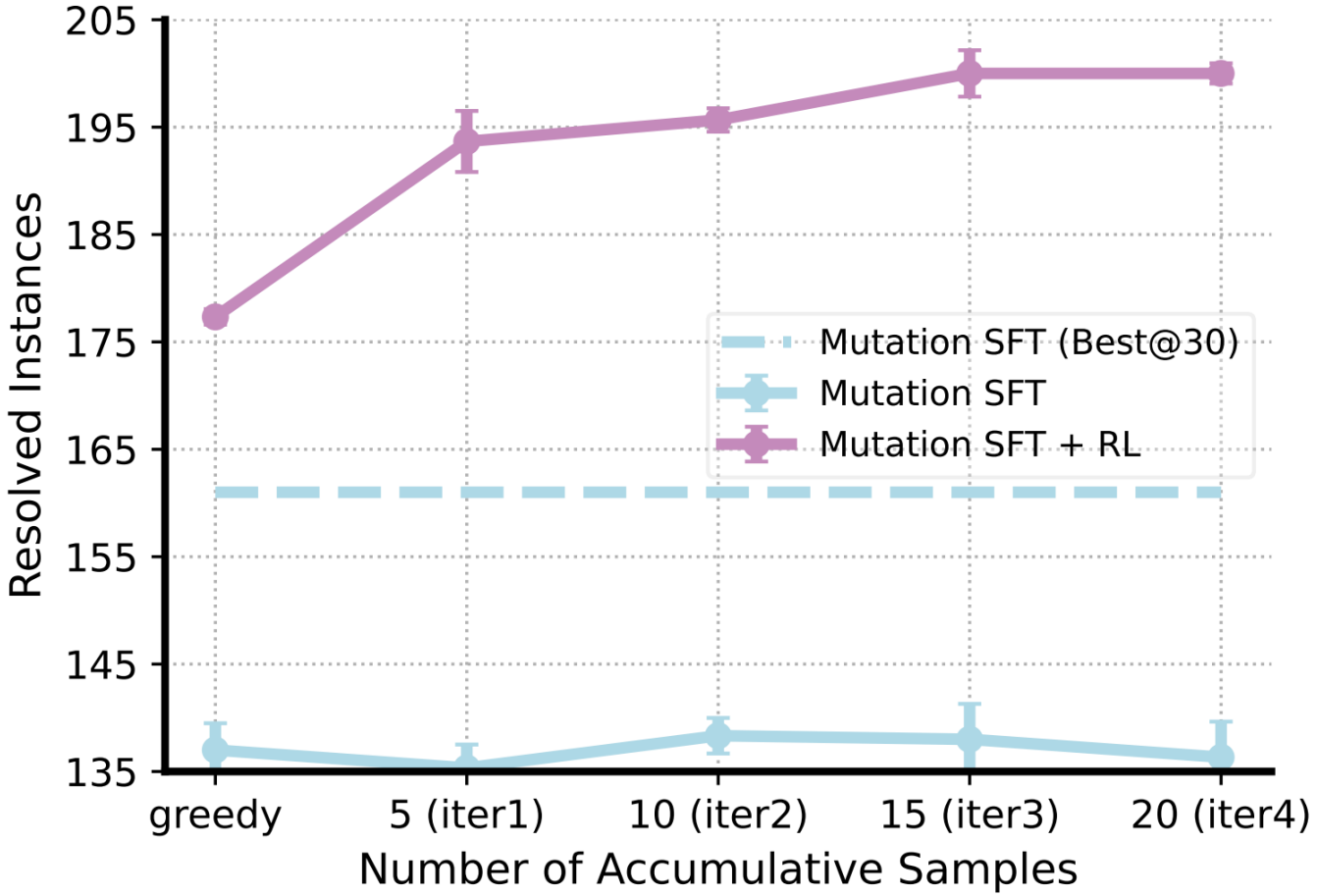
- We adopt potential shaping to alleviate sparse rewards and also teach the model to better evolve.

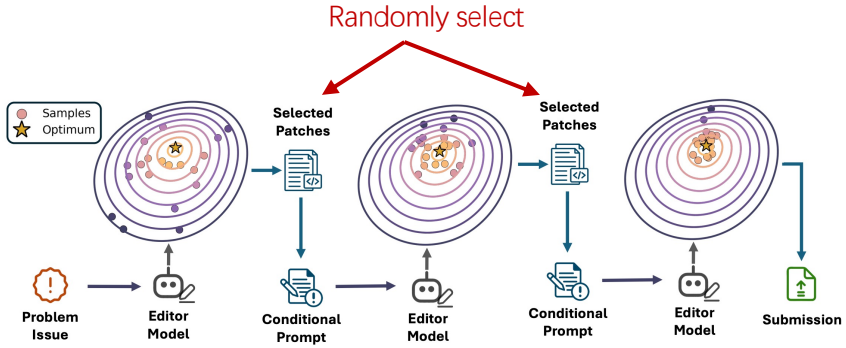
$$r_t = \Phi(y^t) - \Phi(y^{t-1}) = R(x, y^t) - R(x, y^{t-1}).$$

- Therefore, we will have this RL objective:

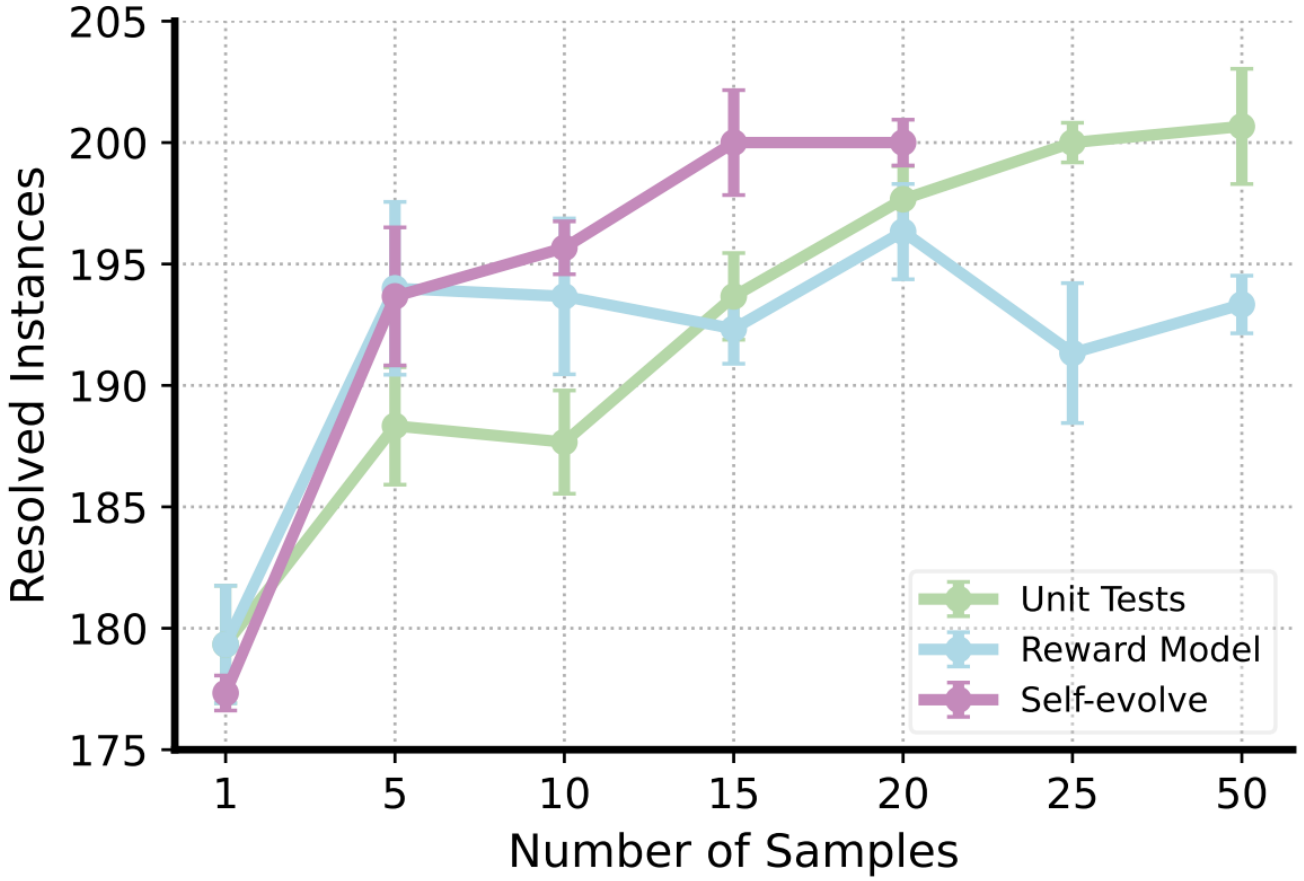
$$\max_{\pi_{\text{RL}}} \mathbb{E}_{y \sim \pi_{\text{RL}}(\cdot|x, C(x), \mathcal{E}), y' \sim \mathcal{E}} \left[R(x, y) - R(x, y') - \lambda F(y) \right].$$

SFT+RL Enables Self-evolve Capability





Evolutionary Test-time Scaling v.s. Other Test-time Scaling



- Parallel Scaling
  - Unit Tests
  - Reward Model
- Evolutionary Scaling
  - Self-evolve

Final Results on SWE-bench Verified

Model Scale	Model/Methods	Scaffold	SWE-Verified Resolved Rate
Large	**GPT-4o** [35]	SWE-agent	23.0
	GPT-4o [33]	Agentless	38.8
	GPT-4o [37]	AutoCodeRover	28.8
	GPT-4o [19]	SWE-SynInfer	31.8
	OpenAI o1 [33]	Agentless	48.0
	Claude 3.5 Sonnet [35]	SWE-agent	33.6
	Claude 3.5 Sonnet [30]	OpenHands	53.0
	Claude 3.5 Sonnet [33]	Agentless	50.8
	Claude 3.5 Sonnet [37]	AutoCodeRover	46.2
	Claude 3.7 Sonnet [1]	SWE-agent	58.2
	DeepSeek-R1 [7]	Agentless	49.2
	DeepSeek-V3 [18]	Agentless	42.0
Small	**Lingma-SWE-GPT-7B (Greedy)** [19]	SWE-SynInfer	18.2
	Lingma-SWE-GPT-72B (Greedy) [19]	SWE-SynInfer	28.8
	SWE-Fixer-72B (Greedy) [34]	SWE-Fixer	30.2
	SWE-Gym-32B (Greedy) [22]	OpenHands	20.6
	SWE-Gym-32B (Best@16) [22]	OpenHands	32.0
	Llama3-SWE-RL-70B (Best@80) [31]	Agentless Mini	37.0
	Llama3-SWE-RL-70B (Best@160) [31]	Agentless Mini	40.0
	Llama3-SWE-RL-70B (Best@500) [31]	Agentless Mini	41.0
	Satori-SWE-32B (Greedy)	Satori-SWE	35.8
	Satori-SWE-32B (Best@10)	Satori-SWE	38.9
	Satori-SWE-32B (Best@25)	Satori-SWE	40.2
	Satori-SWE-32B (Best@50)	Satori-SWE	**41.6**

Thanks