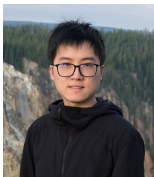


Harness Learning Enables **Generalizable** Test-Time Adaptation



Alvin
Zhang*†



Xuecheng
Liu*



Zixuan
Wang*



Fahim
Tajwar



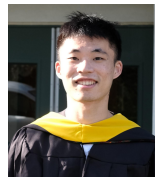
Daman
Arora



Ruslan
Salakhutdinov



Daniel
Khashabi



Yuda
Song‡



Andrea
Zanette‡

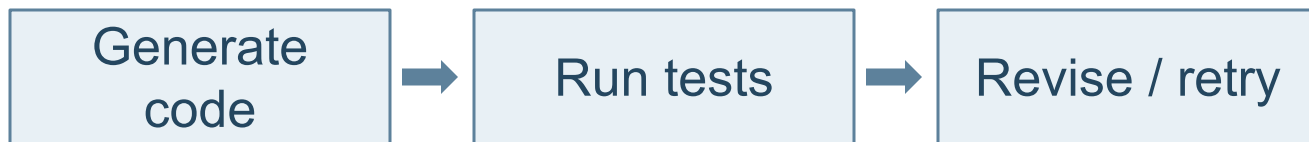
Can experience revising harnesses become a **transferable** skill?

* Equal contribution † Project lead ‡ Equal advising

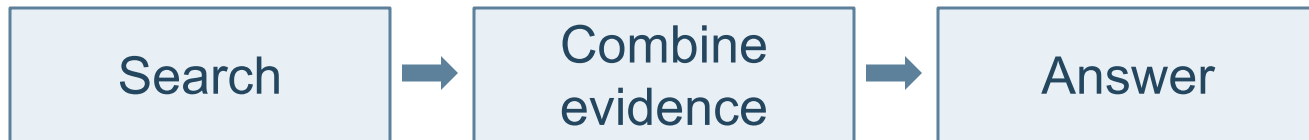
Agent behavior depends on the harness

The harness organizes model calls, tools, and information flow.

Coding



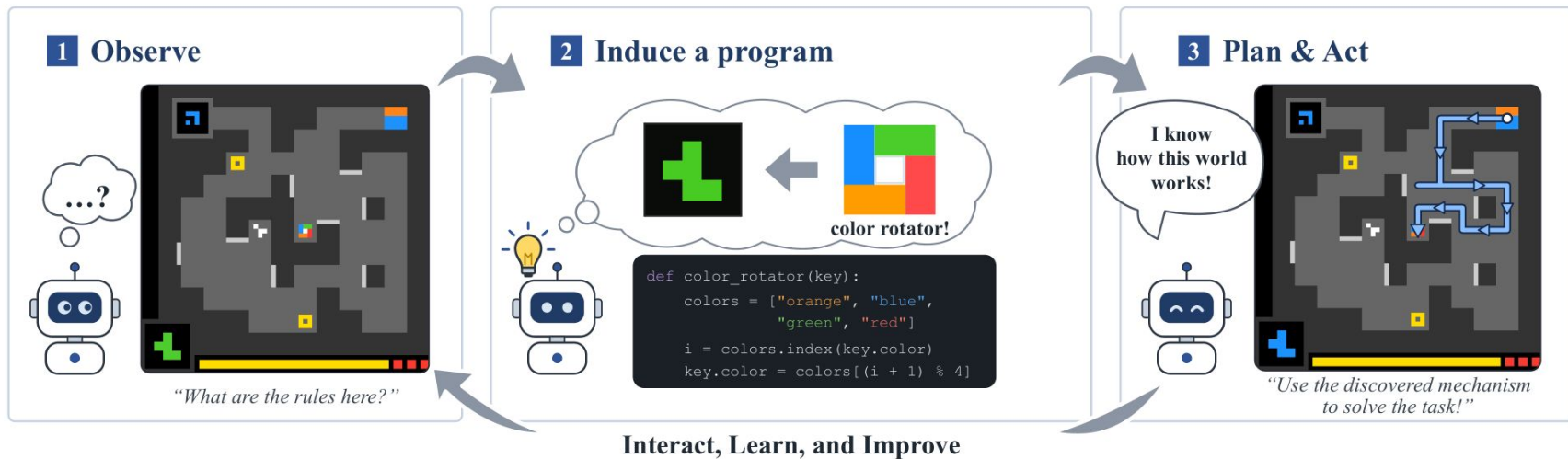
Question answering



A harness that works for one task may need different tools or control flow on another.

Why the harness matters

Schema: discovering the rules of unfamiliar games



Claude Code: 58.7%

Schema: 99.2%

Same Claude Fable 5 model. RHAE (action efficiency) on 25 public ARC-AGI-3 games.

Can we learn to improve harnesses across tasks?

Source: Schema, Zeng et al. (2026), Figure 1 and Section 4.1. Authors' reported results.

Recent work on harness revision

Inference-time revision

TTHE: Test-Time Harness Evolution

Jun Nie et al., 2026

RRSI: Regularized Recursive Self-Improvement of Agent Harnesses

Peng Xia et al., 2026

Training harness models

Harness-R1: Learning to Edit Executable Runtime Harnesses from Agent Failure Trajectories

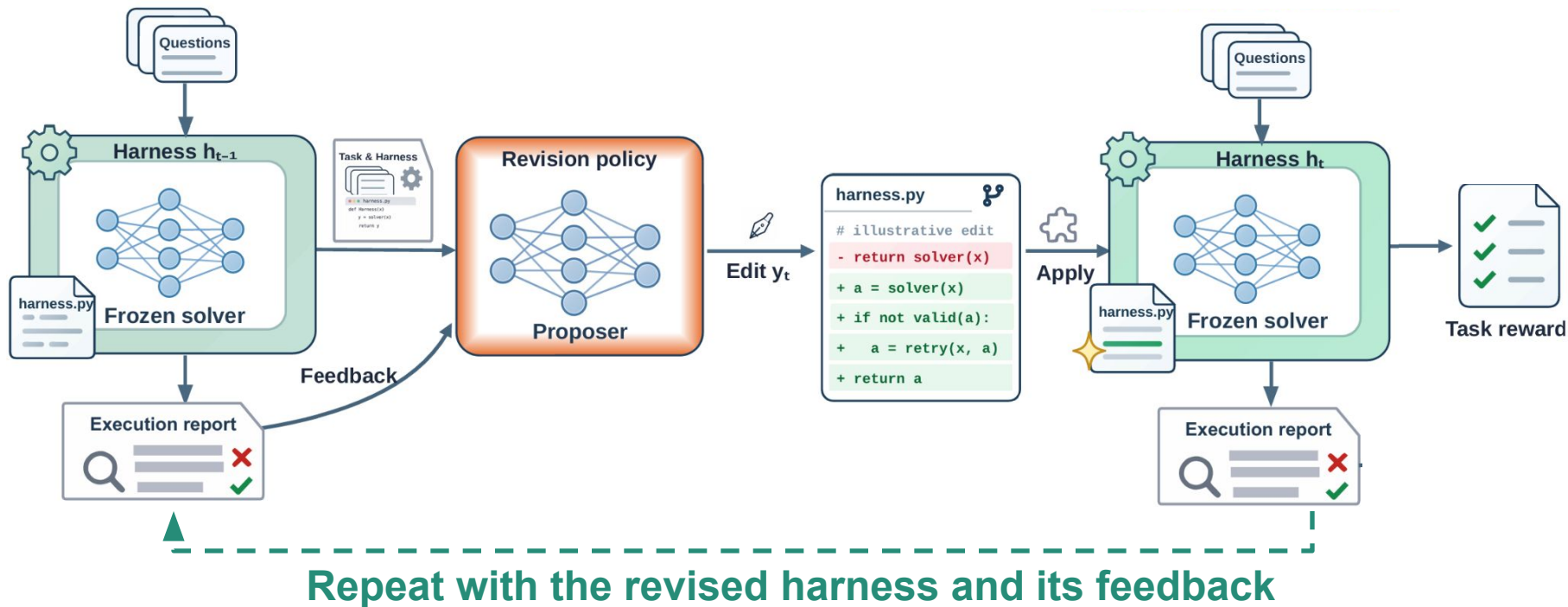
Shuai Shao et al., 2026

JIT-Agent: Scaling Harness Intelligence via Just-in-Time Harness Evolution

JIT-Agent Team, 2026

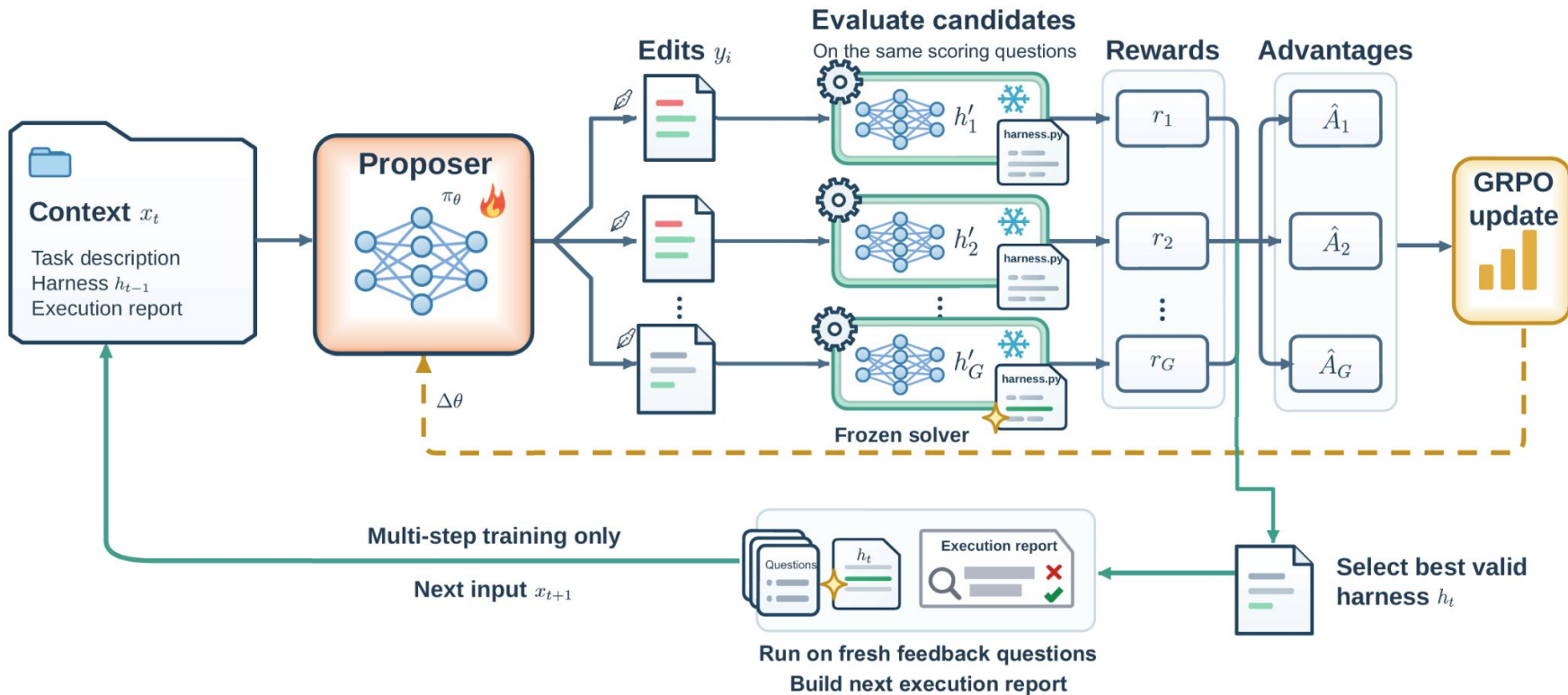
Our meta-learning perspective: train across tasks to learn how to revise harnesses on unseen tasks.

Harness revision at test time



At test time, the proposer and solver weights stay fixed.

Learning from execution outcomes



Evaluating transfer to unseen tasks

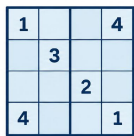
Train the revision model

Reasoning Gym

SFT on 21 families



Maze



Mini Sudoku



Tower of
Hanoi



Color cube

Examples

Then RL on 5 families

HotpotQA



Direct RL

Test on unseen tasks

21 unseen families

Excluded from SFT and RL

Seed: one solver call

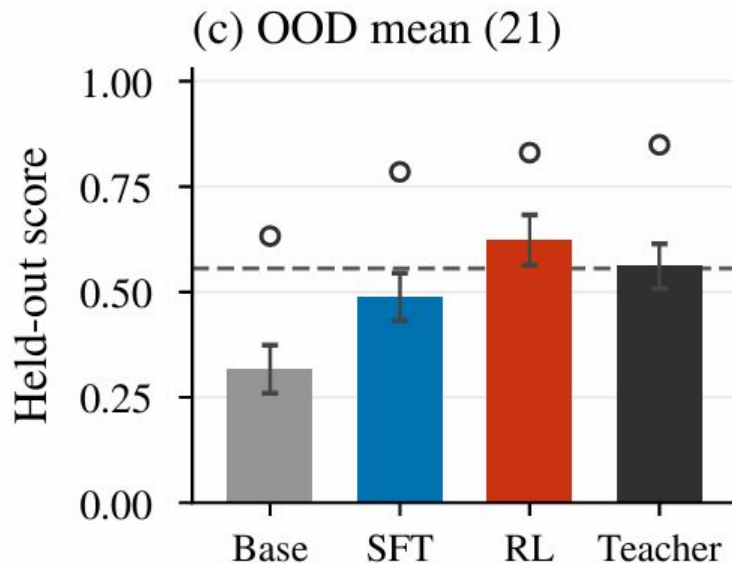
MuSiQue + 2WikiMultihopQA

Seed: retrieval workflow

Separate trained proposers. Fixed proposer and solver weights at test time.

Single revisions generalize to unseen tasks

■ Base ■ SFT ■ Single-step RL — Teacher (35B) -- Seed ○ Best of eight



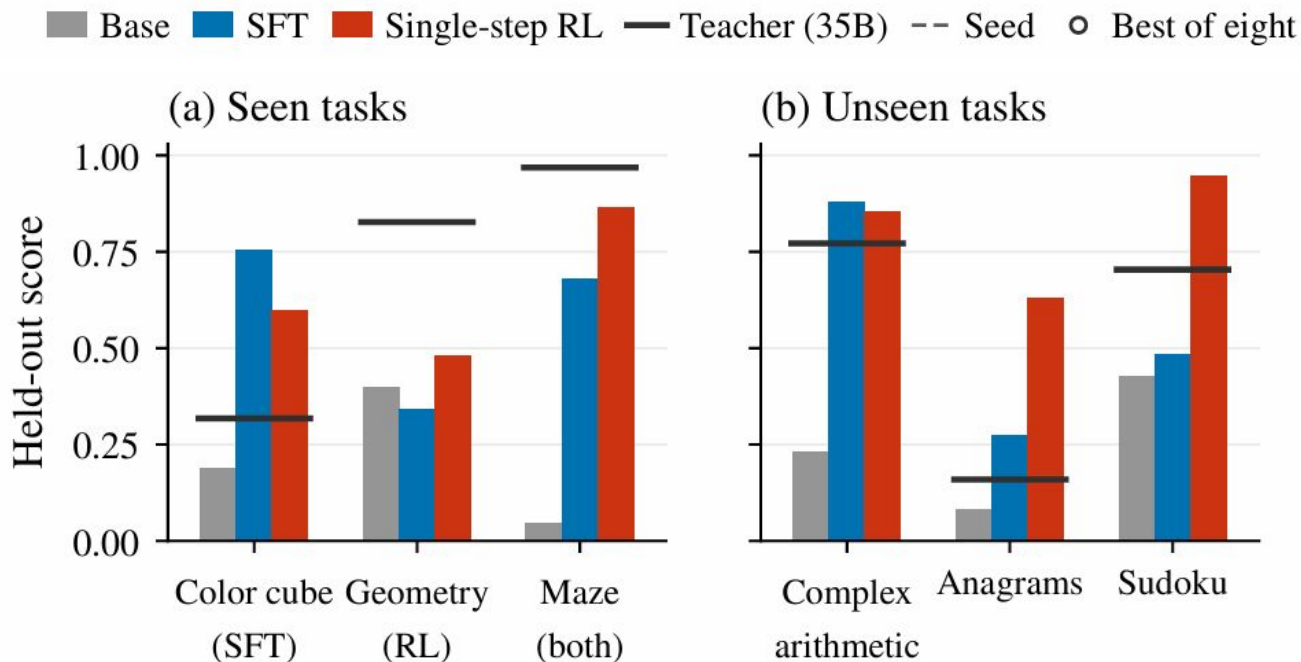
+97%
over Base

Mean revision score

Training improves average revision quality on unseen tasks.
The trained 4B proposer exceeds its 35B teacher on average.

Base means base-model revisions. The teacher remains stronger under oracle best-of-eight selection.

Examples across seen and unseen tasks

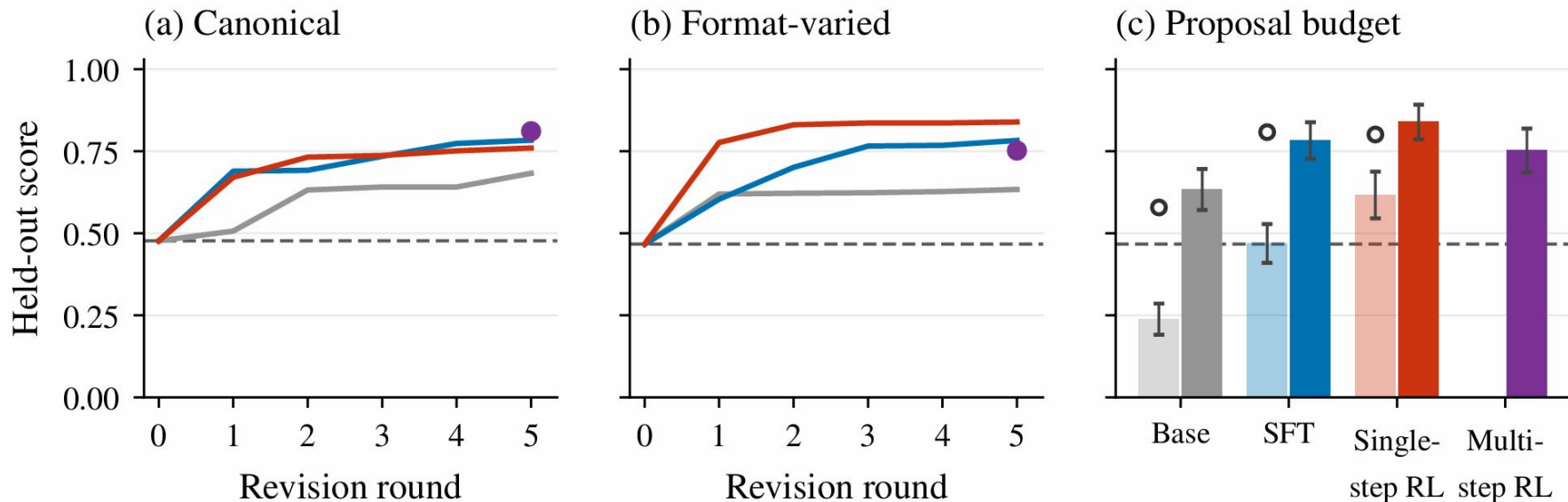


The learned revision skill transfers beyond the families used in training.

Format-varied questions present the same tasks in alternative wording.

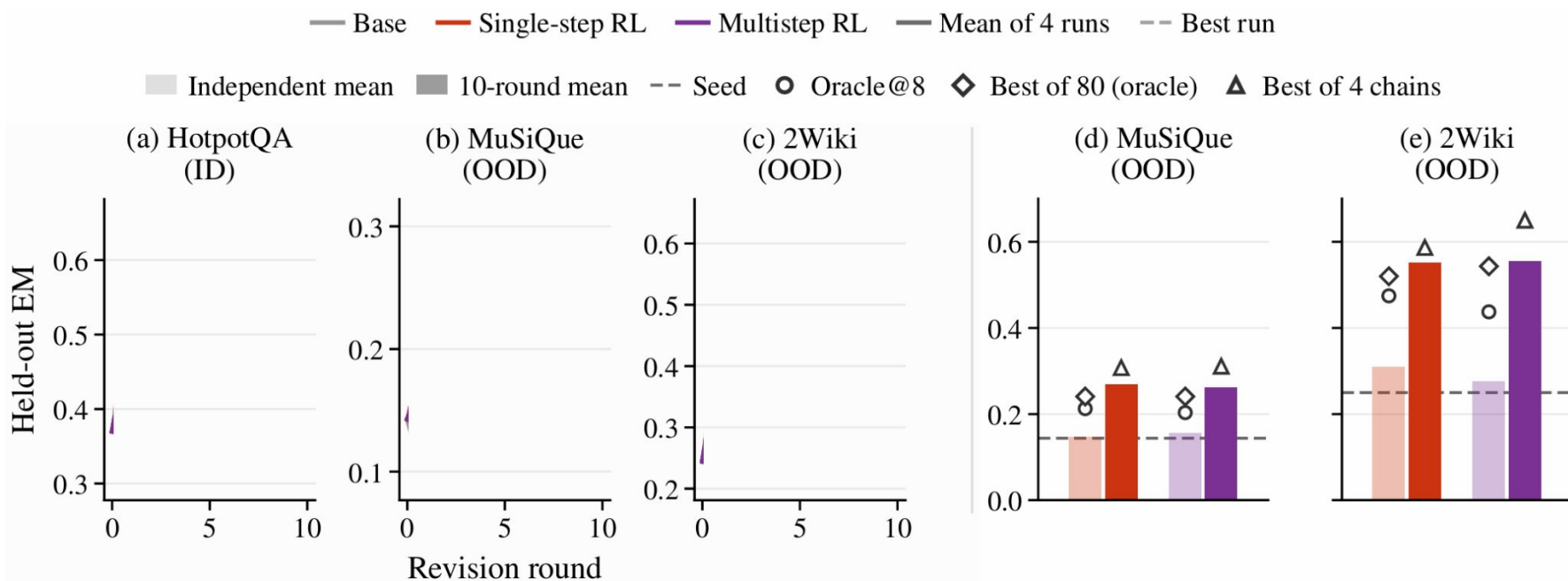
Repeated revision on Reasoning Gym Unseen Tasks

— Base — SFT — Single-step RL — Multistep RL -- Seed ■ Mean of eight ■ Five rounds ○ Best of eight



On format-varied questions, Single-step RL gains arrive mainly in the first round
Five sequential edits approach or exceed oracle best-of-eight selection

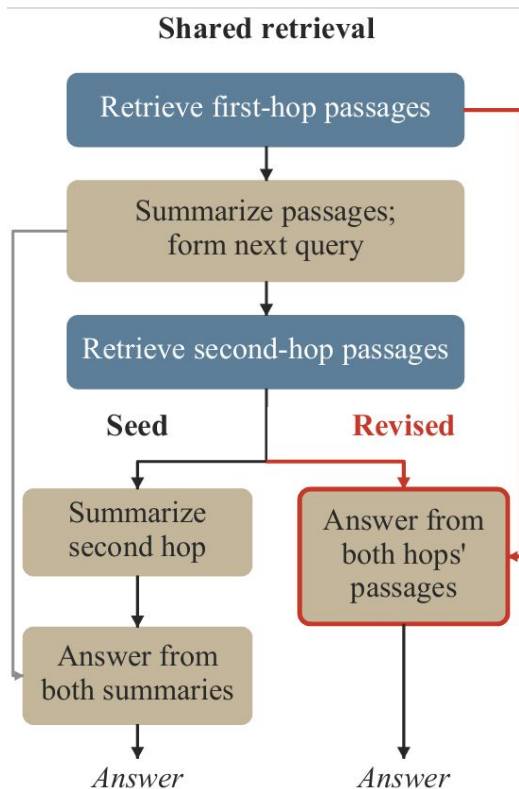
Successive revisions improve unseen QA tasks



Gains continue across rounds on both unseen benchmarks.

Repeated revision outperforms independent proposals at the same proposal budget.

What a learned QA revision changes



Shared retrieval

Summaries guide the next search.

Seed

The final answer uses summaries.

The revised answering call reads the original passages from both hops.

Takeaways

Learn a reusable **harness-revision skill**.

Transfer it to **unseen tasks**.

Improve through **successive revisions**.

The code adapts while model weights stay fixed.

Meta-learning over executable programs

Learn across tasks how to adapt within a task

MAML

Inner loop

Adapt model weights

Harness Learning

Revise harness code

Outer loop: learn across training tasks

Learn a weight initialization by differentiating through inner updates

Train the proposer with GRPO from harness execution rewards

The learned policy adapts code at test time, with both models fixed.

GRPO avoids gradients through code edits and execution.

Limitations and future work

- **Richer training experience:** stronger teachers and demonstrations of successive revisions
- **Agentic proposers:** inspect failures, run targeted tests, and validate edits
- **More demanding settings:** realistic environments and longer-horizon tasks
- **Joint proposer and solver training** with generalization as a central goal

Thank you

Thank you to my collaborators and advisors

Xuecheng Liu Zixuan Wang Fahim Tajwar Daman Arora
Ruslan Salakhutdinov Daniel Khashabi
Yuda Song Andrea Zanette

Paper and code



Training for repeated revision

Repeated revision at test time

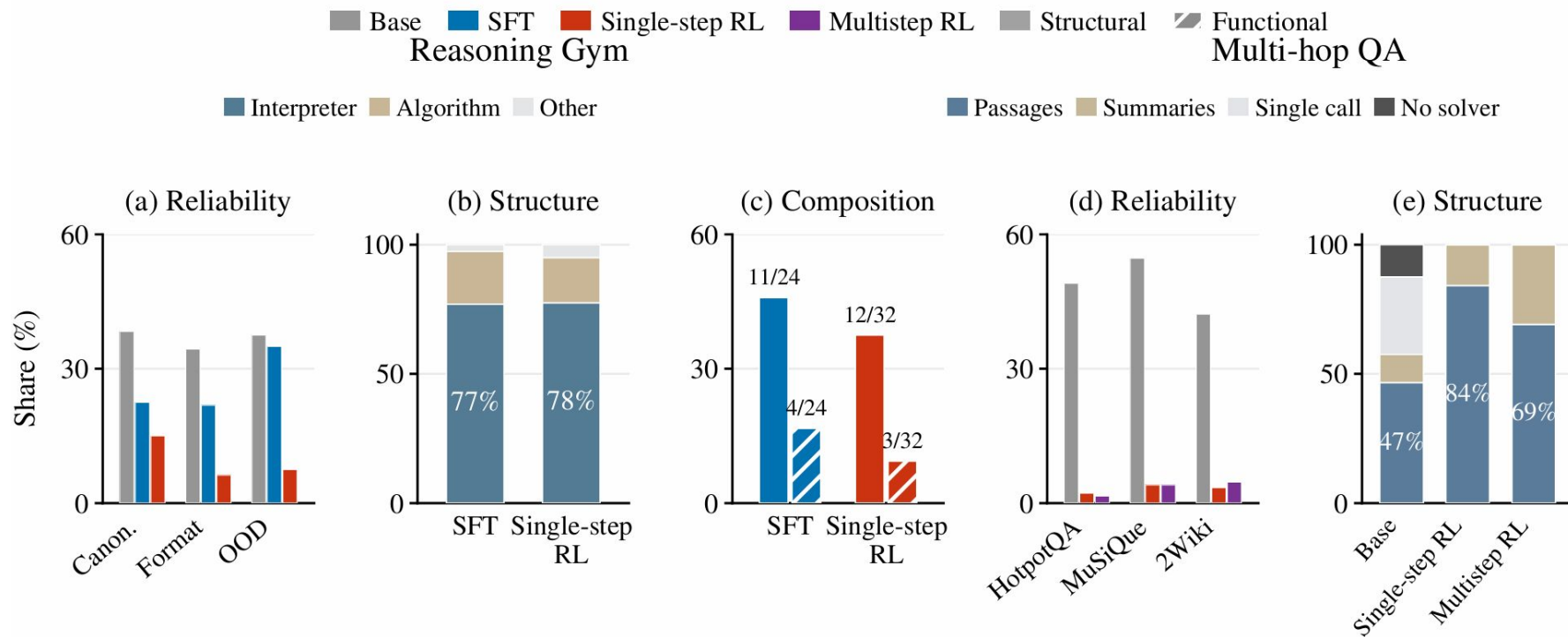
- Single-revision training already supports continued improvement
- In QA, seed-only training transfers to intermediate harnesses and new reports

Training on revision sequences

- Additional gains over single-revision training are not clear
- Rewards remain local to each candidate revision

Open question: can better intermediate data and credit assignment improve long-run adaptation?

Proposal reliability and harness structure



Training produces more viable candidates and changes how harnesses organize computation.