The Interplay of Harness Design and Post-Training in LLM Agents

Kyungmin Kim*, Youngbin Choi*, Seoyeon Lee, Suhyeon Jun, Dongwoo Kim†, Sangdon Park†

Pohang University of Science and Technology (POSTECH)

Studying Harness×Post-Training Beyond Static Env.

The harness is wrapper of LLMs, which sets the system prompt (which tools are exposed or how they're described), auxiliary info, or workflow.

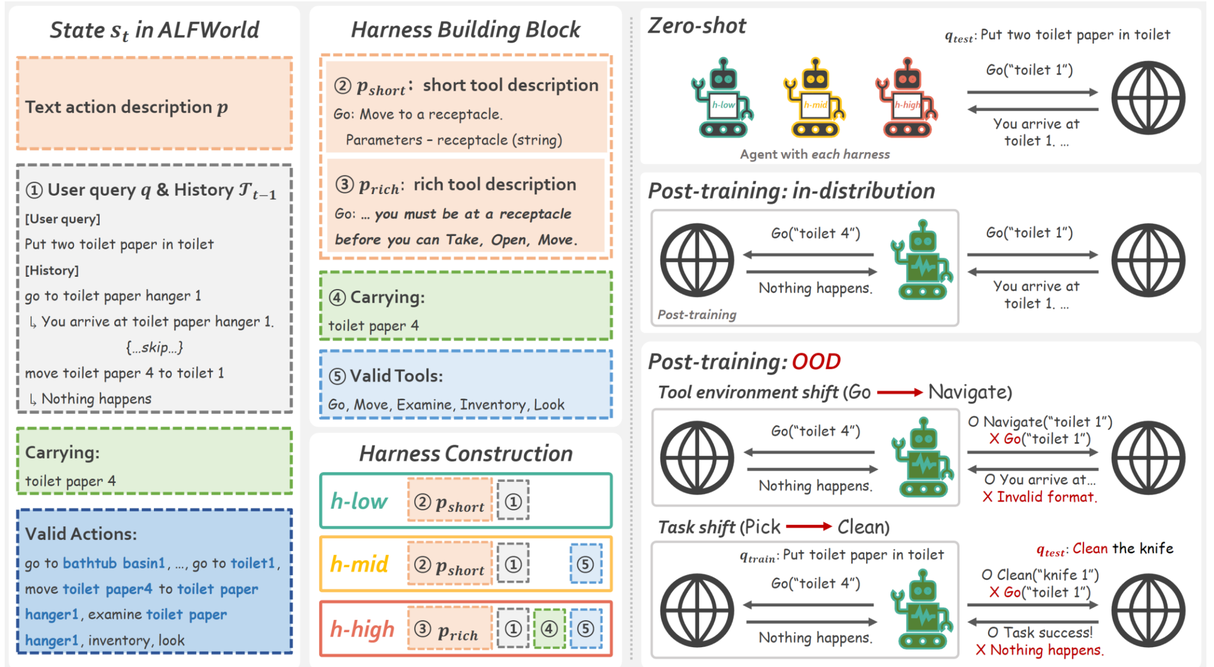
Two blind spots in prior work:

- **The impact of harness design** is only studied during inference time, **not during post-training.**
- **Post-training assumes a static environment** (fixed tools & tasks).

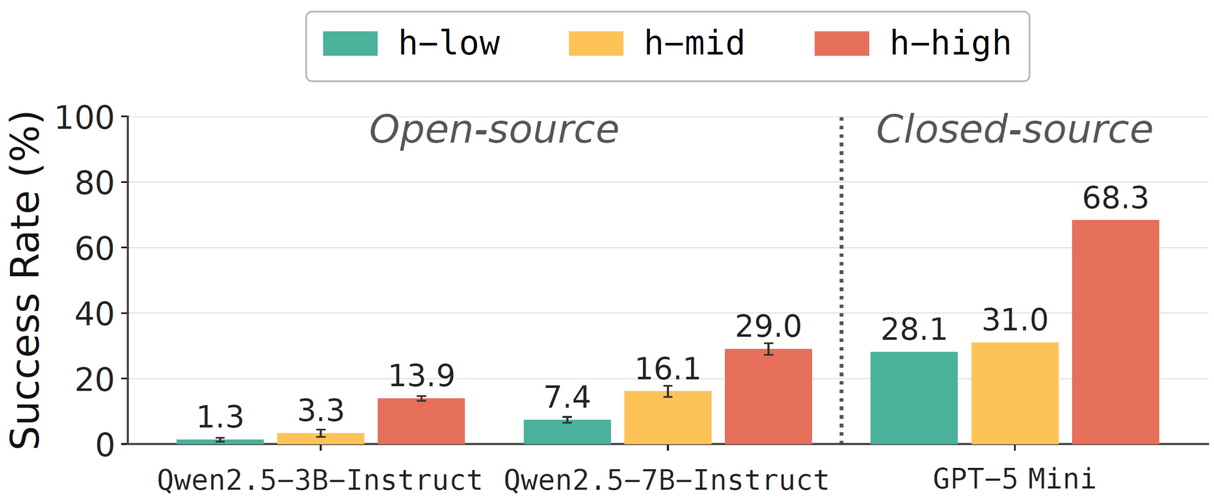
We make the harness a controllable variable and ask:

RQ1. **Does the zero-shot harness gain persist under post-training?**
RQ2. **Does the harness need to be in place during training?**
RQ3. **Does harness-aware post-training give OOD robustness?**

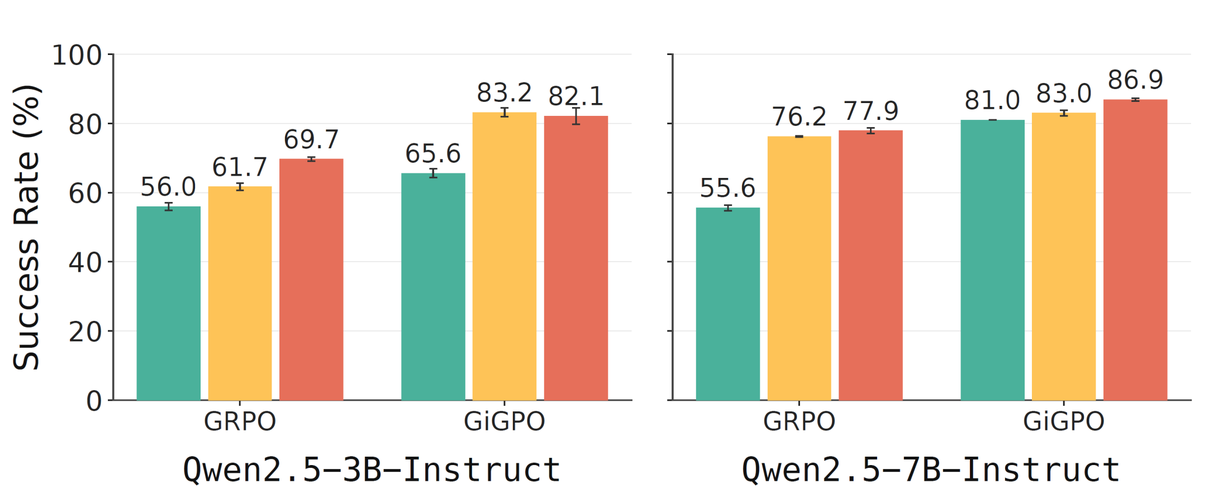
Benchmark: Extension of ALFWrold



RQ1. Harness Gain Under Zero-Shot & Post-Training

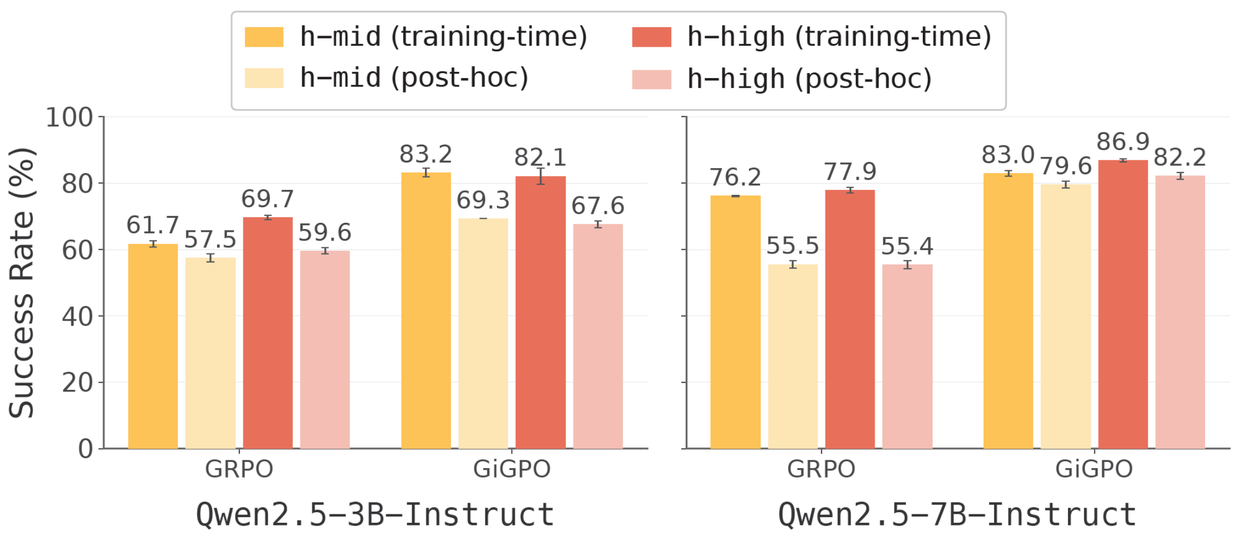


Obs. 1 (Zero-shot): *Harness informativeness monotonically improves zero-shot performance*, ***with larger gains for higher-capacity models.***



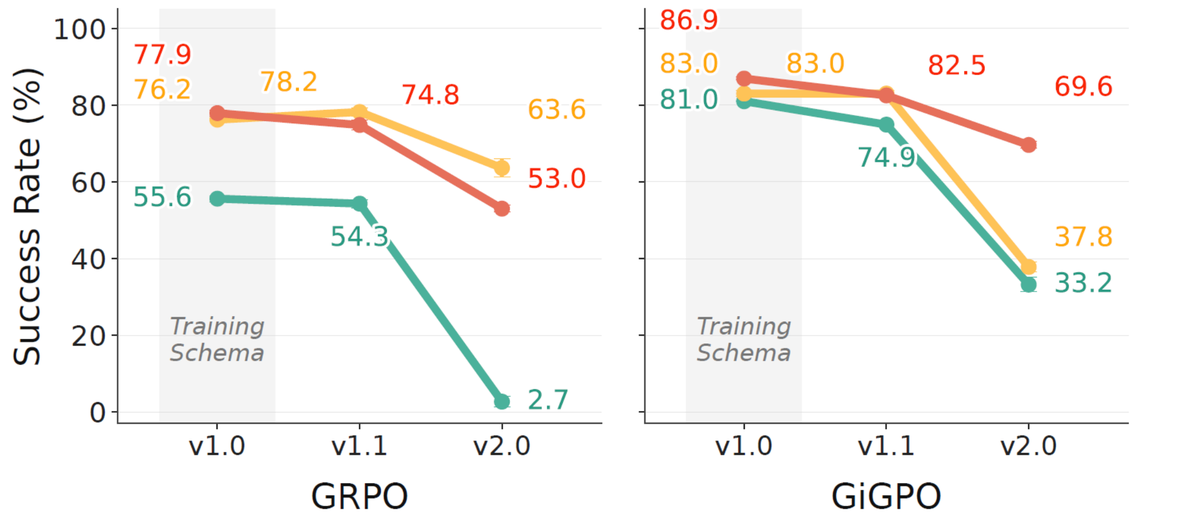
Obs. 2 (In-distribution): *The monotonic harness gain persists* ***after*** *post-training*.

RQ2. Post-Training w Harness vs. Post-Hoc Harness



Obs. 3 (Post-hoc Gap): *Applying a harness only after training* ***recovers*** *little of the benefit of training with it in place*.

RQ3. Post-Training w Harness & OOD Robustness



Harness	Training-time Form (v1.0)	Test-time Form (v2.0)	Agent Output
h-low	Go(receptacle="countertop 1")	ReceptacleControl(action="navigate_to", target="countertop 1")	GoTo("countertop 1")
	Go(receptacle="shelf 1")	ReceptacleControl(action="navigate_to", target="shelf 1")	GoToLocation("shelf 1")
	Go(receptacle="diningtable 1")	ReceptacleControl(action="navigate_to", target="diningtable 1")	GoToDiningTable("diningtable 1")
h-mid	Go(receptacle="countertop 1")	ReceptacleControl(action="navigate_to", target="countertop 1")	navigate_to("countertop 1")
	Go(receptacle="dresser 1")	ReceptacleControl(action="navigate_to", target="dresser 1")	go_to("dresser 1")
	Go(receptacle="dresser 1")	ReceptacleControl(action="navigate_to", target="dresser 1")	goto_dresser()

Obs. 4 (Tool shift): *Harness-aware post-training stays robust to tool shift*; ***h-low collapses under stronger shift.***

Tool Schema & Task Type

We define *tool schema and task type* for OOD *robustness evaluation.*

Tool Schema

Two shift levels on top of v1.0 base (13 tools).

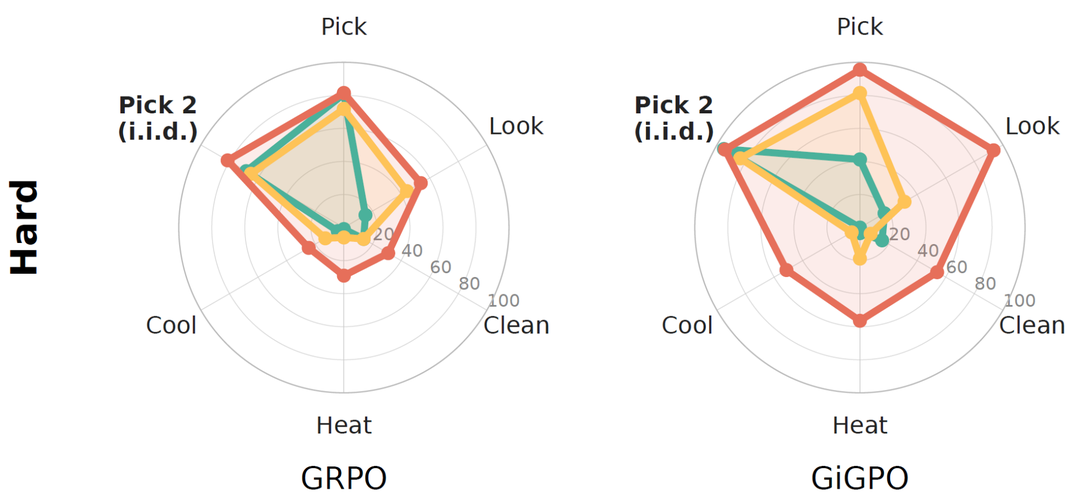
- *v1.1* (mild): Paraphrasing tools and params
- *v2.0* (strong): consolidation of functionally similar tools (13→5), sub-operations exposed via discrete action param

Example
(v1.0) Go(receptacle="drawer 1")
→ ***(v1.1)*** NavigateTo(destination="drawer 1")
→ ***(v2.0)*** ReceptacleControl(action="navigate_to", target="drawer 1")

Task Type

Task categories grouped into *3 difficulty levels by minimum sub-goals.*

- *t-easy*: Pick (4), Look (3)
- *t-med*: Clean/Heat/Cool (5)
- *t-hard*: Pick2 (8)



Obs. 5 (Task shift): *Prior knowledge* ***encoded in harness*** *boosts inter-task transfer* ***of agent performance.***

* Equal Contribution.

† Co-advising.

Paper

The Interplay of Harness Design and Post-Training in LLM Agents

Kyungmin Kim*, Youngbin Choi*, Seoyeon Lee, Suhyeon Jun, Dongwoo Kim†, Sangdon Park†

Pohang University of Science and Technology (POSTECH)

Studying Harness×Post-training Beyond Static Env.

The harness is wrapper of LLMs, which sets the system prompt (which tools are exposed or how they're described), auxiliary info, or workflow.

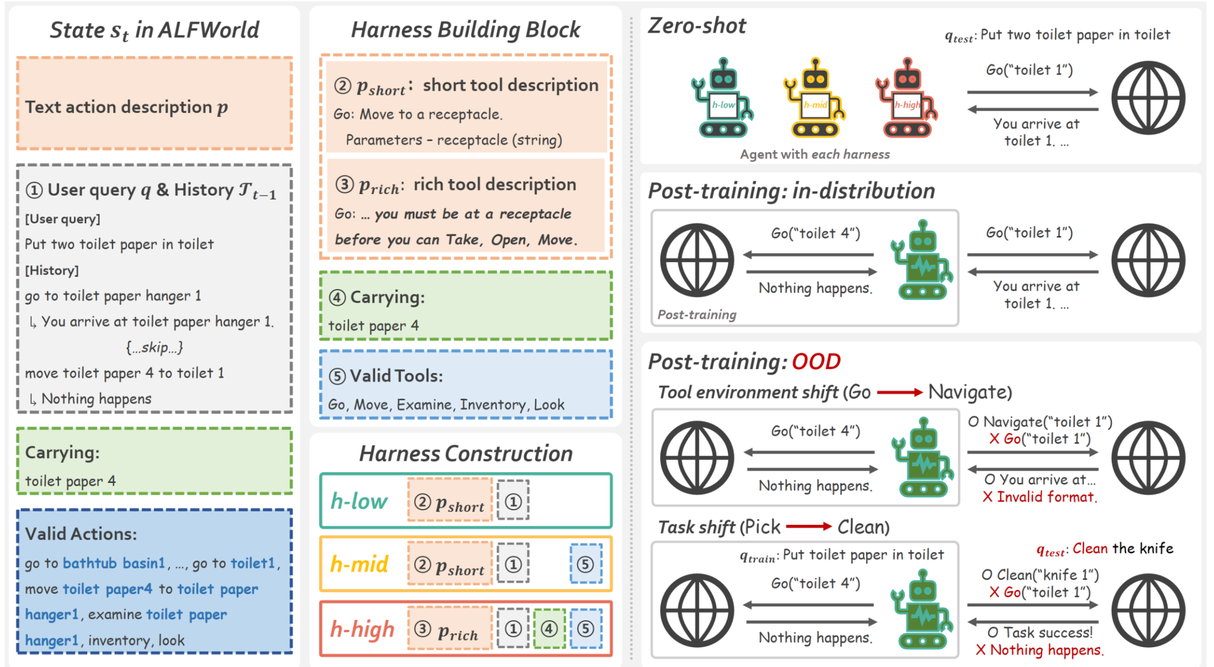
Two blind spots in prior work:

- **The impact of harness design** is only studied during inference time, **not during post-training.**
- **Post-training assumes a static environment** (fixed tools & tasks).

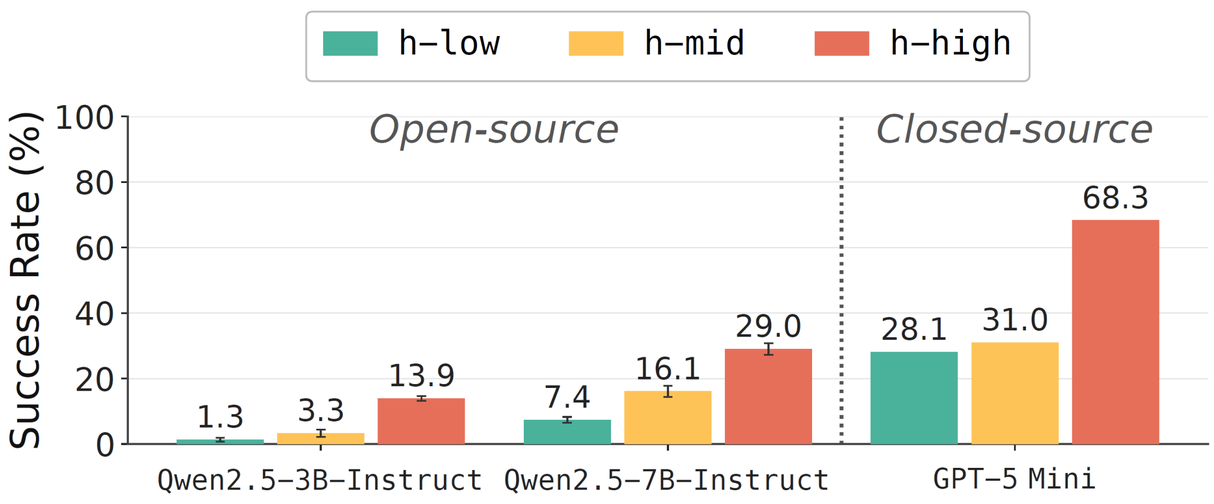
We make the harness a controllable variable and ask:

RQ1. **Does the zero-shot harness gain persist under post-training?**
RQ2. **Does the harness need to be in place during training?**
RQ3. **Does harness-aware post-training give OOD robustness?**

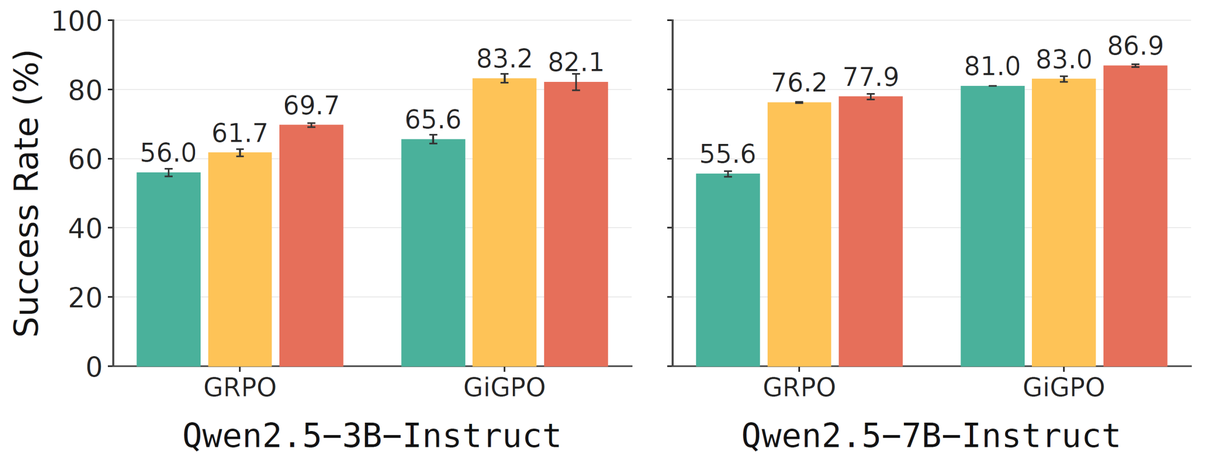
Benchmark: Extension of ALFWrold



RQ1. Does the Zero-Shot Harness Gain Persist Under Post-Training?

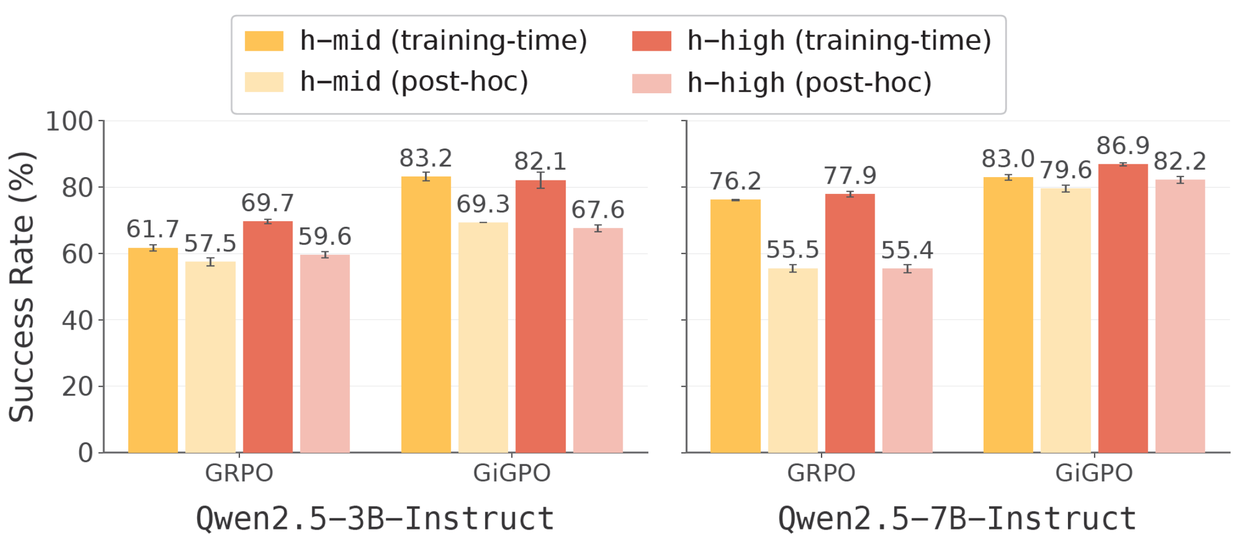


Obs. 1 (Zero-shot): ***Harness informativeness monotonically improves zero-shot performance, with larger gains for higher-capacity models.***



Obs. 2 (In-distribution): ***The monotonic harness gain persists after post-training.***

RQ2. Harness Need to Be In Place During Training?



Obs. 3 (Post-hoc Gap): ***Applying a harness only after training recovers little of the benefit of training with it in place.***

Tool Schema & Task Type

We define ***tool schema and task type*** for OOD ***robustness evaluation.***

Tool Schema

Two shift levels on top of v1.0 base (13 tools).

- *v1.1* (mild): Paraphrasing tools and params
- *v2.0* (strong): consolidation of functionally similar tools (13→5), sub-operations exposed via discrete action param

Example

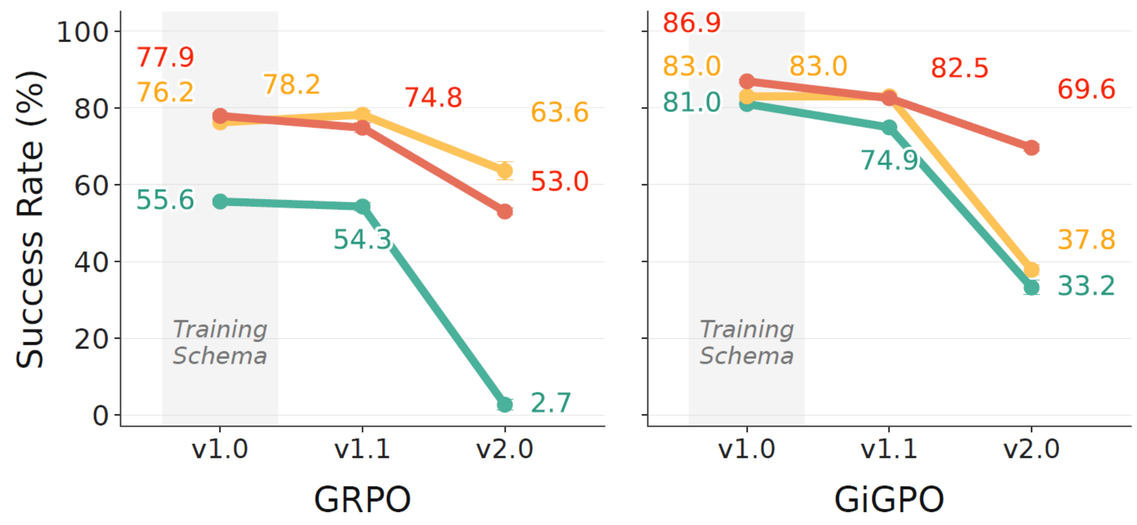
(v1.0) Go(receptacle="drawer 1")
→ ***(v1.1)*** NavigateTo(destination="drawer 1")
→ ***(v2.0)*** ReceptacleControl(action="navigate_to", target="drawer 1")

Task Type

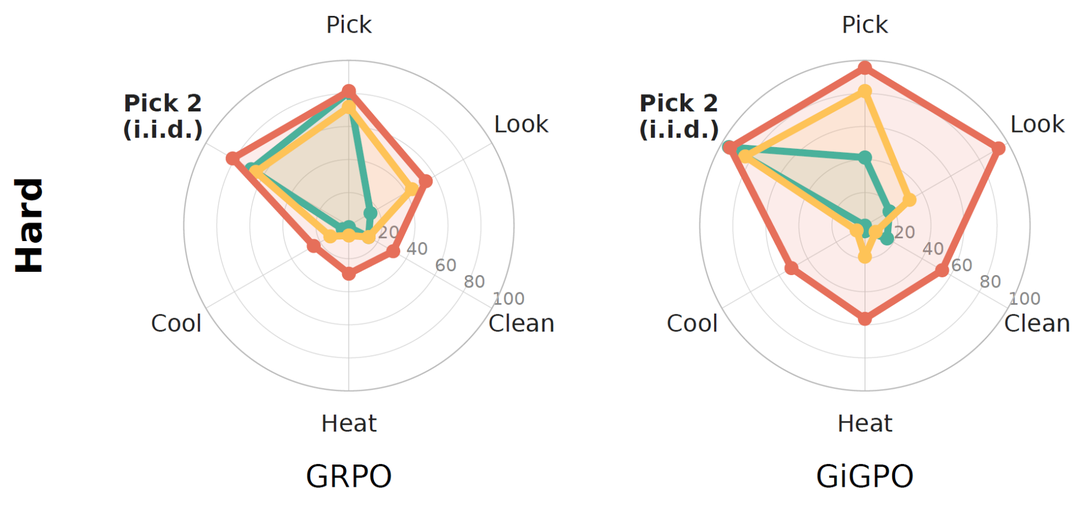
Task categories grouped into ***3 difficulty levels by minimum sub-goals.***

- *t-easy*: Pick (4), Look (3)
- *t-med*: Clean/Heat/Cool (5)
- *t-hard*: Pick2 (8)

RQ3. Does Harness-Aware Post-Training Give OOD Robustness?



Obs. 4 (Tool shift): ***Harness-aware post-training stays robust to tool shift; h-low collapses under stronger shift.***



Obs. 5 (Task shift): ***Prior knowledge encoded in harness boosts inter-task transfer of agent performance.***

* Equal Contribution.

† Co-advising.

Paper