AgentHER: Hindsight Experience Replay for LLM Agent Trajectory Relabeling

Liang Ding
The University of Sydney
liangding.liam@gmail.com

Abstract

LLM-agent training pipelines routinely discard failed trajectories even though GPT-4o achieves only 14–20% on WebArena (Zhou et al., 2024) and below 55% pass@1 on ToolBench (Qin et al., 2024); even specialised systems at 50 -65% (Sodhi et al., 2024; Zhang et al., 2024; Putta et al., 2024) leave the majority of trajectories unused. We introduce **AgentHER**, which recovers this lost signal by adapting *Hindsight Experience Replay* (HER; Andrychowicz et al., 2017) to natural-language agent trajectories: a trajectory that fails goal A is often a *correct* demonstration for an achievable alternative goal B. AgentHER realises this through a four-stage pipeline (failure classification, outcome extraction, LLM-guided relabeling with confidence gating, and data packaging) that converts discarded failures into SFT, DPO, and ShareGPT training data. On WebArena and ToolBench under a strict task-disjoint held-out protocol, AgentHER improves over success-only SFT by **+7.6–11.4%** across four model families (GPT-4o, Qwen2.5-72B/7B, LLaMA-3.1-8B), achieves **2× sample efficiency**, and beats the strongest experience-centric baseline (Agent Workflow Memory) by **+3.0–6.2%**. Two robustness mechanisms—failure-severity weighting and *cross-model* multi-judge verification (gpt-4o-mini paired with Qwen2.5-72B-Instruct)—reduce label noise from 5.9% to 2.9% and raise human-rated relabeling precision to 97.1% on WebArena and 96.0% on ToolBench. A full system-cost audit shows the entire relabeling pipeline costs $2.98 and ≈26 wall-clock minutes for 3,000 trajectories, i.e. $\approx\$1.4\cdot10^{-3}$ per accepted pair. Code: https://github.com/alphadl/AgentHER

1 Introduction

Autonomous LLM agents are deployed across web navigation (Zhou et al., 2024; Deng et al., 2023; Koh et al., 2024), API orchestration (Qin et al., 2024), and simulation (Park et al., 2023; Shen et al., 2023). Despite recent progress, success rates remain modest for the *general-purpose* models that practitioners actually fine-tune: GPT-4o achieves 14–20% on WebArena (Zhou et al., 2024), and ToolBench pass@1 remains below 55% (Qin et al., 2024). Heavily engineered, multi-stage agents push WebArena toward 50–65% (Sodhi et al., 2024; Zhang et al., 2024; Putta et al., 2024), but those gains come from *inference-time* composition (prompt stacks, tree search, workflow memory) on top of the same underlying LLMs; the underlying training pipelines still discard the majority of collected trajectories, leaving the dominant source of experience untouched.

The data-waste problem. Standard pipelines (Agent-Tuning (Zeng et al., 2024), FireAct (Chen et al., 2023), Agent-FLAN (Chen et al., 2024)) discard failed trajectories (Figure 1), wasting 60–75% of collected data. Failures are not random noise: they are coherent executions that frequently achieve correct intermediate results under the wrong goal. An agent that searches seven suppliers and identifies one just above a price cap has done a *complete, correct* comparison—ideal training signal for a re-framed goal.

Connection to HER. HER (Andrychowicz et al., 2017) converts sparse-reward RL failures by substituting the intended goal with the one actually achieved. Lifting HER to LLM agents requires understanding what was achieved across multi-step tool interactions and synthesising a natural-language prompt the trajectory genuinely satisfies.

AgentHER. AgentHER automates hindsight relabeling via a four-stage pipeline (Figure 2) and adds two mechanisms to control label quality: **failure-severity weighting** (drops trajectories with major reasoning flaws, inspired by MQM error analysis) and **cross-model multi-judge verification** (two independently trained judges—gpt-4o-mini and Qwen2.5-72B-Instruct—must both accept a relabeling), which together cut label noise from 5.9% to 2.9%.

Contributions.

- An offline pipeline adapting goal-conditioned HER to natural-language agent trajectories, producing SFT/DPO/ShareGPT datasets without extra environment interactions (distinct from inference-time approaches such as ECHO Hu et al., 2025 and AWM Wang et al., 2024).
- Failure-severity weighting and cross-model multi-judge verification raise relabeling precision from

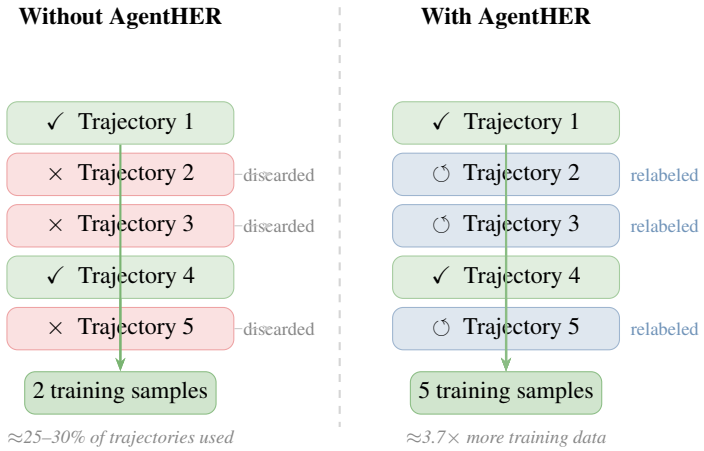

Figure 1: **AgentHER vs. conventional pipelines.** Standard training retains only successes (✓), discarding 60–75% of data. AgentHER relabels failures (↻) with achievable hindsight goals, expanding the effective training corpus $\approx 3.7\times$ at a cost of $\approx \$1.4 \cdot 10^{-3}$ per accepted pair (Section 4.3).

94.1% to 97.1% on WebArena and 96.0% on ToolBench (Section 5.3).

- +7.6–11.4% over SFT-Success on a strict task-disjoint held-out split; +3.0–6.2% over the strongest baseline (AWM); gains hold across 1.5B–72B parameters (+5.6–9.0%) and compound under iterative redeployment (+10.4% over four rounds).

- A full cost audit: $2.98 and 26 wall-clock minutes for 3,000 trajectories (3.27–4.27 LLM calls per trajectory), confirming that sample efficiency gains are not offset by pipeline overhead.

2 Related Work

Hindsight Experience Replay. HER (Andrychowicz et al., 2017) relabels sparse-reward RL failures by substituting the goal with the one actually reached; the goal-conditioned formulation traces back to Kaelbling (1993) and was extended to deep networks (Sutton et al., 1999; Mnih et al., 2015; Precup et al., 2000). Lifting HER to LLM agents adds two requirements absent in prior RL formulations: it must determine what was actually achieved across unstructured multi-step tool interactions, and it must synthesise a natural-language prompt the trajectory satisfies—not merely substitute a vector-space goal. Existing failure-reuse work in NLP (negative-example filtering, contrastive augmentation) selects or reweights existing (goal, trajectory) pairs but does not *generate* a new achievable goal.

LM-agent adaptations of HER. Hu et al. (2025) propose ECHO, a prompting framework that adapts HER for *online* LM agents via a scratchpad memory that allows arbitrary rewriting of both goals and intermediate steps—a runtime planner. AgentHER differs in two ways: it is *offline* training-data augmentation, and it relabels only the *goal* while keeping the trajectory unchanged, producing SFT/DPO/ShareGPT datasets. We compare with an offline-faithful re-implementation of ECHO in Section 4.2.

Experience-centric agent learning. ExpeL (Zhao et al., 2024) extracts experience rules at inference time without modifying weights; ETO (Song et al., 2024) performs DPO over *matched* success/failure pairs of the *same* task; AWM (Wang et al., 2024) induces and reuses workflow patterns from *successful* trajectories; Reflexion (Shinn et al., 2023) applies verbal RL online. A parallel line—FireAct (Chen et al., 2023), Agent-Tuning (Zeng et al., 2024), Agent-FLAN (Chen et al., 2024), AgentGym (Xi et al., 2024)—fine-tunes on curated successes only. AgentHER requires none of this: it converts existing failures offline without extra interactions or matched successes, and produces standard SFT/DPO data that can be combined with any of the above. We compare with ExpeL, ETO, and AWM under identical infrastructure in Section 4.2.

Self-improvement, verification, and critique. Synthetic data (Wang et al., 2023b; Peng et al., 2023), reward-filtered self-training (Gulcehre et al., 2023), self-verified reasoning (Zelikman et al., 2022), and preference optimisation (Rafailov et al., 2023) all operate on single-step text or matched preferences, not on multi-step agent trajectories. For verification, CRITIC (Gou et al., 2024) uses tool-augmented critics, process reward models (Lightman et al., 2024) score reasoning steps, and self-consistency (Wang et al., 2023a) aggregates *same-model* samples. Our multi-judge follows a multi-agent debate (Du et al., 2024; Liang et al., 2023) instead: we require independence of *both* model and decoding, with an explicit ablation in Section 3.5.

Test-time compute and inference scaling. A complementary axis improves agent performance through *repeated inference* rather than better training data. Levi (2024) provide a statistical model of inference scaling, showing that pass@k improves predictably with the

number of trials: given per-sample failure probability p_i, pass@k $= 1 - \frac{1}{n}\sum_i p_i^k$ follows a power-law decay in an "inference loss" as k grows. AgentHER operates on the orthogonal dimension: by converting failures into high-quality training data it reduces the inherent p_i for each task—the floor that inference scaling subsequently compounds over. The two mechanisms are therefore complementary: AgentHER strengthens the base model offline; test-time compute then extracts additional coverage from that stronger starting point.

3 The AgentHER Framework

3.1 Problem Formulation

Let $\mathcal{F} = \{(g_i, \tau_i, f_i)\}_{i=1}^{N}$ be a corpus of N failed agent runs, where $g_i \in \mathcal{G}$ is the natural-language goal, f_i is a failure label, and $\tau_i = \big((z_t^i, a_t^i, o_t^i)\big)_{t=1}^{T_i}$ is the thought–action–observation sequence. Let $\mathcal{S} = \{(g_j^+, \tau_j^+)\}_{j=1}^{M}$ be available successful demonstrations. **Goal**: produce high-quality training pairs $\{(\hat{g}_i, \tau_i)\}$ from $\mathcal{F}$ by synthesising *hindsight goals* $\hat{g}_i$ for which τ_i is a valid positive demonstration.

Definition 3.1 (Valid Hindsight Goal)**.** *A prompt $\hat{g} \in \mathcal{G}$ is a* valid hindsight goal *for trajectory τ if:* (a) *every factual claim implied by $\hat{g}$ is supported by the observations $\{o_t\}$, and* (b) *two independent LLM judges $\mathcal{J}_1, \mathcal{J}_2$ each assign confidence $c_k(\hat{g}, \tau) \geq \theta$ that τ is a successful demonstration of $\hat{g}$.*

The mapping $\Phi : \mathcal{F} \to \mathcal{D} \cup \{\perp\}$ assigns each failed run a training datum $(\hat{g}_i, \tau_i)$ or rejects it ($\perp$). The augmented corpus $\mathcal{S} \cup \{\Phi(\cdot) \neq \perp\}$ grows from M to up to $M + N$ labelled pairs.

3.2 Pipeline Overview

The pipeline processes each failed trajectory in four stages. Stages 1 and 2 offer rule-based (free) and LLM-judge variants; Stage 3 requires *two* independent LLM calls in multi-judge mode; Stage 4 is deterministic. Algorithm 1 gives the end-to-end procedure.

3.3 Stage 1: Failure Detector

The Failure Detector assigns each trajectory a failure type $\phi \in \Phi_{\text{types}}$, a recoverability flag $r \in \{0, 1\}$, and a **severity weight** $w \in [0, 1]$. $\Phi_{\text{types}} = \{$INCOMPLETE, CONSTRAINT_VIOLATION, WRONG_RESULT, TOOL_ERROR, HALLUCINATION, OFF_TOPIC$\}$. The taxonomy was defined before data collection, drawing on NLP error typologies (Lu et al., 2024) and WebArena task semantics, and applied uniformly by a single classifier.

Rule-based mode aggregates trajectory text and matches per-type keyword lexicons. Severity score: $v = \min(1.0,\ 0.3 + 0.1 \cdot h)$, h = matched keywords. Recoverability: $r = 1$ iff at least one observation exceeds minimum length *and* $\phi \neq$ TOOL_ERROR.

LLM-judge mode presents the full trajectory with a JSON-schema output prompt requesting $(\phi, v, r, w, \text{explanation})$.

Algorithm 1 AgentHER End-to-End Relabeling

Require: Failed corpus $\mathcal{F} = \{(g_i, \tau_i, f_i)\}$, threshold θ, max retries K, severity threshold δ, relabeler $\mathcal{J}_1$, verifier $\mathcal{J}_2$
Ensure: Augmented dataset $\mathcal{D}^+$

$\mathcal{D}^+ \leftarrow \emptyset$
for each $(g_i, \tau_i, f_i) \in \mathcal{F}$ **do**
  $(\phi_i, r_i, w_i) \leftarrow$ FAILUREDETECTOR(τ_i) ▷ type, recoverability, severity weight
  if $r_i = 0$ **or** $w_i < \delta$ **then**
    continue ▷ discard irrecoverable or major-flaw runs
  end if
  outcome$_i \leftarrow$ OUTCOMEEXTRACTOR(τ_i)
  $\hat{g}_i^*, c^* \leftarrow$ NONE, 0
  for $k = 1, \ldots, K$ **do**
    $(\hat{g}, b, c_1) \leftarrow \mathcal{J}_1(\text{outcome}_i, g_i)$
    if $b = 1$ **and** $c_1 \geq \theta$ **then**
      $c_2 \leftarrow \mathcal{J}_2(\hat{g}, \tau_i)$ ▷ cross-model verification
      if $c_2 \geq \theta$ **then**
        $\hat{g}_i^* \leftarrow \hat{g}$; $c^* \leftarrow (c_1 + c_2)/2$
        break
      end if
    else if $b = 1$ **and** $c_1 > c^*$ **then**
      $\hat{g}_i^* \leftarrow \hat{g}$; $c^* \leftarrow c_1$
    end if
  end for
  if $\hat{g}_i^* \neq$ NONE **and** $c^* \geq 0.8\theta$ **then**
    datum $\leftarrow$ DATAAUGMENTER$(\hat{g}_i^*, \tau_i, w_i)$
    $\mathcal{D}^+ \leftarrow \mathcal{D}^+ \cup \{\text{datum}\}$
  end if
end for
return $\mathcal{D}^+$

Severity weighting. Inspired by MQM-style error analysis in machine-translation evaluation (Lu et al., 2024), we distinguish *major* errors (reasoning contradictions, hallucinated observations, catastrophic tool misuse) from *minor* errors (constraint violations, incomplete results). Major-error trajectories receive $w < \delta = 0.3$ and are discarded; minor-error trajectories receive $w \in [0.3, 1.0]$ and are passed downstream, with w used to weight the DPO loss at Stage 4. This reduces the fraction of noisy relabelings from 5.9% to 2.9% (Table 3).

3.4 Stage 2: Outcome Extractor

Stage 2 produces a REPLAYOUTCOME: a list of actual achievements and key observations (numeric data, entity names, facts) that anchor Stage 3 and prevent hallucinated hindsight prompts.

Rule-based: each non-trivial, non-error observation is treated as one achievement (truncated to 200 chars); numeric tokens extracted by regex. **LLM**: richer summaries handling implicit results, deduplication, and strict factuality—only facts evidenced by observations.

3.5 Stage 3: Cross-Model Relabeling and Verification

Given the REPLAYOUTCOME and original prompt for style reference, the *relabeler* $\mathcal{J}_1$ synthesises:

$$\hat{g},\ b_{\text{valid}},\ r_{\text{rationale}},\ c_1 \longleftarrow \mathcal{J}_1(\text{outcome},\ g_{\text{orig}}) \tag{1}$$

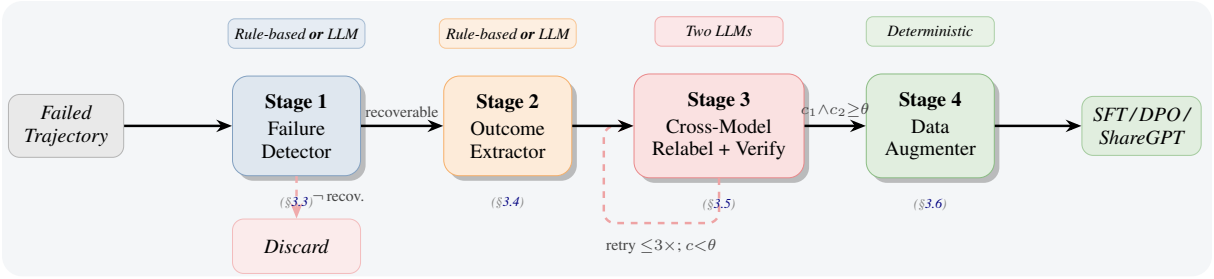

Figure 2: **AgentHER four-stage pipeline.** Stage 1 classifies failures and discards irrecoverable runs (dashed downward arrow). Stage 3 retries up to three times if the relabeling confidence $c < \theta$. Stage 3 uses two independently trained LLMs (gpt-4o-mini and Qwen2.5-72B-Instruct) for relabeling and verification; both must agree (Section 3.5). Stages 1–2 offer zero-cost rule-based variants; Stage 4 is deterministic.

with $b_{\text{valid}} \in \{0, 1\}$, $c_1 \in [0, 1]$. Four constraints: *(1)* $\hat{g}$ reads as a natural user request; *(2)* every assertion is satisfied by observations; *(3)* $\hat{g}$ does not reference the original failed prompt; *(4)* complexity matches g_{orig}.

Cross-model multi-judge verification. Calling the *same* LLM twice with the same prompt at T=0 provides no genuine statistical independence. Our multi-judge protocol therefore uses two independently trained judges: $\mathcal{J}_1$ (`gpt-4o-mini`, closed-weight; T=0.3 first attempt, T=0.7 retries) and $\mathcal{J}_2$ (`Qwen2.5-72B-Instruct`, open-weight via vLLM (Qwen Team, 2025); T=0, schema-constrained decoding). Because the two models have different architectures and training corpora, their agreement reflects genuine inter-model consensus rather than intra-model self-consistency (Wang et al., 2023a). Acceptance requires both $c_1 \geq \theta$ *and* $c_2 \geq \theta$. We refer to this configuration as **AgentHER-MJ-X** (cross-model). For ablation we also report: **AgentHER-MJ-S** (same-model, Qwen2.5-72B used twice with different temperatures) and the original **AgentHER-SJ** (single-judge $\mathcal{J}_1$ only). The cross-model variant is the default throughout the rest of the paper unless stated otherwise.

Validation loop. Up to $K = 3$ relabeler attempts; the first attempt that passes both judges is accepted. A best-effort fallback is retained if no attempt passes the multi-judge bar but at least one passes the single-judge bar at $c_1 \geq 0.8\theta$. Full prompt templates for all four stages are provided in Appendix B.

3.6 Stage 4: Data Augmenter

Stage 4 serialises each accepted $(\hat{g}, \tau)$ pair into three formats:

- **SFT**: a two-turn conversation $[(\texttt{user}, \hat{g}), (\texttt{assistant}, \tilde{a})]$ where $\tilde{a}$ reconstructs the agent's chain of thought from τ; loss scaled by severity weight w.
- **DPO**: chosen $(\hat{g}, \tau)$ paired with rejected (g_{orig}, τ), encoding the preference that $\hat{g}$ is the correct goal; w scales the reward margin (formula in Appendix C).
- **ShareGPT**: multi-turn format compatible with LLaMA-Factory, ms-swift, and FastChat.

3.7 Theoretical Plausibility

The proposition below is a plausibility argument rather than a deployment guarantee: it shows that accepted pairs are unbiased under a perfect judge and gives a noise-tolerance bound for imperfect ones. The full proof is in Appendix D.7.

Proposition 3.1 (Unbiasedness under a perfect judge)**.** *Let $\pi^*_{\mathcal{G}}$ be the oracle goal-conditioned policy. Under a* perfect *judge $\mathcal{J}$ ($c(\hat{g}, \tau) = 1 \Leftrightarrow \tau$ is a valid demonstration of $\hat{g}$), every accepted pair $(\hat{g}_i, \tau_i)$ is a correct (goal, trajectory) sample from the support of $\pi^*_{\mathcal{G}}$.*

Corollary 3.1.1 (Noisy-judge bound)**.** *With an imperfect judge of precision p, the expected gain over SFT-Success is lower-bounded by $p \cdot \Delta_{perfect} - (1 - p) \cdot \varepsilon$, where ε bounds the marginal harm of a single noisy pair. For our cross-model multi-judge precision $p = 0.971$, this bound is positive whenever $\varepsilon \leq 33 \cdot \Delta_{perfect}$; for the single-judge bound ($p = 0.941$), it requires $\varepsilon \leq 16 \cdot \Delta_{perfect}$.*

An empirical sanity check grounding the bound against our held-out results is in Appendix D.7.

Goal-distribution analysis. A potential concern is whether AgentHER inadvertently amplifies high-frequency task types while neglecting low-frequency behaviours (Ding et al., 2022). We address this empirically in Section 5.4.

4 Experiments

4.1 Experimental Setup

Benchmarks and a strict task-disjoint split. WebArena (Zhou et al., 2024): 812 compositional web-automation tasks (Shopping, Reddit, GitLab, Maps, Wikipedia). We use a *strict task-disjoint* split: WA-TRAIN (612 tasks, collection only) and WA-HELDOUT (200 tasks, evaluation only), stratified by category (seed 0, released as JSON). We collect 3,000 failed and 500 successful trajectories from WA-TRAIN; acceptance rates: SJ 78.0%, MJ-X 71.5%. Full-task numbers are in Appendix D.1.

ToolBench (Qin et al., 2024): 16,464 tool-use tasks across 49 APIs (G1/G2/G3 splits). We collect 5,000 failed and 2,000 successful trajectories from the official training splits; acceptance rates: SJ 82.5%, MJ-X 75.5%.

A weaker collector (GPT-3.5-turbo) yields richer failures; the judges are stronger and serve no collection role (Appendix A).

Train/test integrity. Fine-tuned models are independent of the GPT-3.5-turbo collector; training data encodes what it did on WA-TRAIN, not solutions to WA-HELDOUT. WebArena environments are reset between collection and evaluation. **SFT-Random** uses the same 3,000 failures as AgentHER (no relabeling), making it a rigorous equal-volume control. All results are averaged over 3 random seeds; std $<$0.5 pts (Appendix E).

Models and fine-tuning. We evaluate **GPT-4o** (OpenAI, 2023), **Qwen2.5-72B/7B** (Qwen Team, 2025), and **LLaMA-3.1-8B** (Dubey et al., 2024). Open-weight models use LoRA (Hu et al., 2022) (rank 16, α=32, 3 epochs, 8×A100-80G; Appendix A). We also report **GPT-4o-ICL** (8 relabeled exemplars prepended, no fine-tuning) as a reproducible alternative that does not depend on the OpenAI fine-tuning API.

Baselines. Training-time baselines: **Base** (zero-shot), **SFT-Random** (same 3k failures as AgentHER, no relabeling; equal-volume control), **Rejection-Sampling** (filter c>0.5, no relabeling), **SFT-Success** (successes only), **SFT-Negative** ("the following attempt failed" prepended; equal volume), **ETO** (Song et al., 2024) (offline DPO over matched success/failure pairs of the *same* task), **ECHO-offline** (Hu et al., 2025) (offline re-implementation materialising ECHO's trajectory rewrites into SFT data), and **AWM** (Wang et al., 2024) (workflow-memory induction from successes, combined with SFT-Success). Inference-time reference only (†, weights unchanged): **Reflexion** (Shinn et al., 2023), **ExpeL** (Zhao et al., 2024). AgentHER variants: **SJ** (single-judge), **MJ-S** (same-model, ablation), **MJ-X** (cross-model, default). SFT-Success is the Δ reference; AgentTuning and FireAct score below it under our protocol (Appendix D.1).

Metrics. WebArena: success rate (%) on WA-HELDOUT. ToolBench: pass@1 (%) on G1/2/3 splits.

4.2 Main Results

Discussion. AgentHER-MJ-X improves over SFT-Success by 7.6–8.7% on WebArena and 7.6–11.4% on ToolBench on the task-disjoint split, only 0.3–1.3% below full-task numbers (App. D.1). It beats every experience-centric baseline by +3.0–6.2%: ETO requires matched success/failure pairs per task, ECHO-offline rewrites trajectories and can amplify reasoning errors, and AWM builds workflow memory from successes only. AgentHER bypasses all three constraints.

The cross-model judge (MJ-X) outperforms the single-judge (SJ) by +0.8–1.6% and the same-model analogue (MJ-S) by +0.5%, confirming that model-level independence matters more than temperature diversity alone. Smaller models benefit more: 7B/8B gains (+10.7–11.4% on ToolBench) exceed Qwen-72B (+8.1%) and GPT-4o (+7.6%), consistent with smaller models having more room to gain from richer training signal. SFT-Negative provides only +0.4–0.8% over SFT-Random, well below AgentHER's +12–18% (WebArena: +12–13%; ToolBench: +14–18%), because appending a failure marker still discards the positive content that hindsight relabeling recovers. AgentHER-MJ-X trained on WA-TRAIN achieves +9.5% over SFT-Success when evaluated zero-shot on ToolBench (Tab. 12), suggesting what transfers is planning ability rather than memorised task descriptions.

4.3 System Cost Analysis

Table 2 addresses the "sample-efficiency vs. system-efficiency" confound directly. Per accepted pair, MJ-X costs $\$1.4{\cdot}10^{-3}$ ($\approx\$1.4$ per 1,000 pairs); the multi-judge overhead over SJ is $+\$1.20$ (+67%) and $+9.7$ min (+61%), repaid by a 94.1→97.1% precision boost and a +0.8–1.6% downstream gain. Of the combined relabeling+LoRA budget ($7.18 / 4.8 h), relabeling accounts for $\approx$9% of wall-clock but $\approx$42% of dollars: LoRA dominates wall-clock time while LLM API spend dominates marginal cost. The 2× sample efficiency (Section 4.4) means AgentHER saves the cost of collecting 50% of successful demonstrations from scratch—fresh rollouts are orders of magnitude more expensive per pair than a relabeling call. Stage-level token breakdown: App. F.

4.4 Data Efficiency and Scaling

Figure 3(a) shows AgentHER-SJ reaches full-SFT-Success performance with only 50% of successful demos (2× data efficiency); AgentHER-MJ-X exceeds it at every data point. Panel (b) confirms log-linear scaling with failure volume—unlike SFT-Success which cannot leverage additional failures. Panel (c) shows the optimal threshold θ^*=0.5 is consistent across both variants, making it a robust hyperparameter.

4.5 Model-Size Scaling

All five Qwen2.5 instruction-tuned checkpoints (1.5B–72B) are evaluated under identical training conditions (Figure 4). All three methods improve monotonically with scale. AgentHER-MJ-X gains peak at 14B (+9.0%) and dip slightly at 72B (+7.8%), suggesting larger models are already partially saturated. The 1.5B model nearly doubles in performance (6.4% → 12.0%), making AgentHER viable for edge deployment.

4.6 Ablation Study

Key findings. Cross-model multi-judge (MJ-X) beats same-model multi-judge (MJ-S) by +0.5% and cuts label noise from 4.4% to 2.9%; decoding-temperature independence alone (MJ-S vs. SJ: +0.3%) is less valuable than model-level independence. Severity weighting

Table 1: **Main results.** WebArena: success rate (%) on the strict task-disjoint WA-HELDOUT split. ToolBench: pass@1 (%) on official test splits. Best per column in **bold**; second-best underlined. Δ vs. SFT-Success; Δ_b vs. best training-time baseline (AWM). All fine-tuned results averaged over 3 seeds; std <0.5 pts (Appendix E). †Inference-time, online; reference only. ‡Same 3,000 failed trajectories as AgentHER (equal-volume control). ICL = 8-shot in-context, no fine-tuning.

	WebArena WA-HELDOUT (%)				ToolBench pass@1 (%)			
Method	GPT-4o	Qwen-72B	Qwen-7B	LLaMA-8B	GPT-4o	Qwen-72B	Qwen-7B	LLaMA-8B
Base	13.7	20.9	8.2	6.5	53.2	60.1	44.6	42.1
SFT-Random‡	18.0	25.6	13.9	12.7	61.4	68.9	54.8	52.9
SFT-Negative	18.6	26.1	14.5	13.1	62.1	69.5	55.6	53.7
Rejection-Sampling	20.7	28.6	16.9	15.5	66.2	73.7	59.8	57.4
SFT-Success	22.3	29.9	17.8	16.4	67.8	75.4	61.2	58.3
ETO	24.4	32.1	21.0	19.4	71.0	78.5	64.8	61.5
ECHO-offline	24.8	32.5	21.6	19.9	71.6	79.1	65.5	62.1
AWM	25.6	33.4	22.4	20.6	72.4	79.8	66.4	62.9
Reflexion†	*18.4*	*27.3*	*15.6*	*14.2*	*63.8*	*71.6*	*56.8*	*55.2*
ExpeL†	*19.5*	*28.8*	*18.4*	*16.9*	*65.0*	*73.0*	*60.4*	*57.6*
GPT-4o-ICL	28.4	—	—	—	73.2	—	—	—
AgentHER-SJ	28.7	36.9	25.7	24.0	73.9	82.6	71.0	67.5
AgentHER-MJ-X	**29.9**	**37.7**	**26.5**	**24.8**	**75.4**	**83.5**	**72.6**	**69.0**
Δ vs. SFT-Succ.	+7.6	+7.8	+8.7	+8.4	+7.6	+8.1	+11.4	+10.7
Δ_b vs. AWM	+4.3	+4.3	+4.1	+4.2	+3.0	+3.7	+6.2	+6.1

Table 2: **System-cost audit** on 3,000 WebArena failures. S1–S3a shared by SJ/MJ-X; only S3b verifier is added in MJ-X. Pricing assumes gpt-4o-mini Batch API + prompt caching; Qwen-72B amortised on 8×A100. Per-stage I/O token sizes are in App. F. *LoRA cost: 4.4 h × 8 GPUs × $1.05/h.

Stage	**Calls**/tr.	**Cost** ($)	**Wall** (m)
S1 detector	1.00	0.39	4.1
S2 extractor	1.00	0.55	4.6
S3a relabeler	1.27	0.84	6.7
S3b verifier *(MJ-X only)*	1.00	1.20	10.1
S4 augmenter (det.)	—	0.00	0.4
SJ total	3.27	1.78	15.8
per accepted (78.0%)	4.19	$7.6{\cdot}10^{-4}$	—
MJ-X total	4.27	2.98	25.9
per accepted (71.5%)	5.97	$1.4{\cdot}10^{-3}$	—
LoRA fine-tune Qwen-7B	—	4.20*	263
End-to-end MJ-X	—	7.18	289

accounts for -1.5%, confirming that discarding major-flaw trajectories is worth the reduced yield. Confidence filtering is the single most critical component: removing it drops 4.2% and raises noise to 15.0%. Naive relabeling harms performance by -6.1%, showing that random prompt assignment cannot substitute for goal reverse-engineering. Finally, the DPO preference signal contributes -2.5% over SFT, indicating the rejected/chosen pairing adds value beyond positive supervision alone.

4.7 Qualitative Examples

Appendix G shows two annotated relabeling examples—both CONSTRAINT_VIOLATION failures, one from WebArena and one from ToolBench—including original failed goals, trajectories, accepted hindsight goals, and cross-model judge rationales.

5 Analysis

5.1 Failure-Mode Distributions on Both Benchmarks

Table 4 reports the failure-type distribution on both benchmarks. The failure-mode mix differs sharply: ToolBench has 6× the rate of TOOL_ERRORs, reflecting unreliable remote API endpoints, while WebArena has 1.7× the rate of CONSTRAINT_VIOLATIONs, reflecting under-specified UI tasks. "Looping" trajectories make up 12.1% of WebArena failures; Stage 1 rejects 76% of them outright (no useful intermediate state), and the remaining 24% still yield a positive +3.4% Δ over SFT-Success—below non-looping ($\Delta = +9.5\%$) but above the discard-all baseline.

5.2 Per-Failure-Type Improvement on Both Benchmarks

INCOMPLETE (+11.0/12.7% on WA/TB) and CONSTRAINT_VIOLATION (+9.6/11.0%) benefit the most: these trajectories contain rich, factually correct observations misaligned with the original goal. TOOL_ERROR benefits the least (+2.0/1.8%) because crashes leave minimal signal regardless of benchmark. On WebArena the two highest-gain types account for ≈63% of all failures; on ToolBench they account for only ≈38%—explaining why ToolBench gains are larger in absolute % (since lower-baseline categories also see meaningful gain) but more dependent on AgentHER's ability to extract value from WRONG_RESULT and TOOL_ERROR.

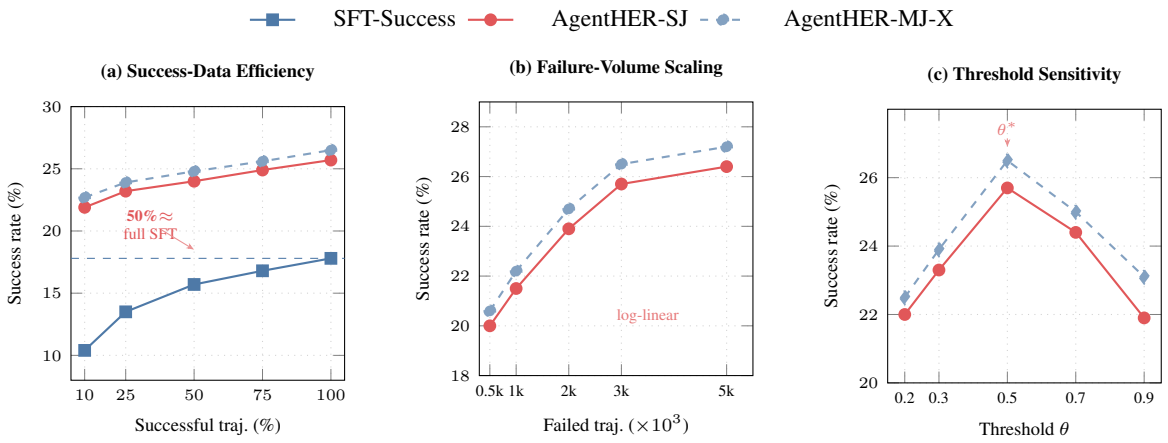

Figure 3: **Data efficiency and scaling** (Qwen2.5-7B, WA-HELDOUT). *(a)* AgentHER-SJ at 50% successful demos matches SFT-Success at 100%; AgentHER-MJ-X exceeds it. *(b)* Both AgentHER variants scale log-linearly with failure volume; SFT-Success cannot benefit from additional failures. *(c)* Both variants peak near θ^*=0.5; MJ-X is uniformly better and slightly more robust at low θ.

Table 3: **Component ablation** on WA-HELDOUT (Qwen2.5-7B). Δ relative to Full AgentHER-MJ-X; noise = fraction of accepted relabelings rated invalid by human annotators (Section 5.3). "MJ-S" = same-model multi-judge (Qwen-72B used twice with different temperatures, an analogue of self-consistency). "MJ-X" = cross-model multi-judge (gpt-4o-mini + Qwen-72B-Instruct), our default.

Configuration	Success (%)	Δ	Noise (%)
Full AgentHER-MJ-X (cross-model judge + severity)	**26.5**	—	2.9
w/ Same-model multi-judge (MJ-S, two temps)	26.0	−0.5	4.4
w/o Multi-judge (single judge only, AgentHER-SJ)	25.7	−0.8	5.9
w/o Severity weighting (w=1 for all)	25.0	−1.5	4.3
w/ Rule-based Extractor (Stage 2 degraded)	24.6	−1.9	6.0
w/ LLM Detector (Stage 1 upgraded)	27.4	+0.9	2.4
w/o Failure Detection (accept all)	23.8	−2.7	7.5
w/o Confidence Filter (θ=0)	22.3	−4.2	15.0
SFT only (no DPO, relabeled as positives)	24.0	−2.5	2.9
Naive Relabeling (random prompt)	20.4	−6.1	n/a
SFT-Success (no AgentHER)	17.8	−8.7	—

Table 4: **Failure-type distribution** of collected failed trajectories. WebArena (3,000 trajectories) and ToolBench (5,000 trajectories), classified by Stage 1. "Looping" is a sub-mode of INCOMPLETE where the agent repeats one action ≥ 3 times; reported separately given its distinct recoverability profile.

Failure type	WebArena	ToolBench
INCOMPLETE	32.4%	21.0%
of which *looping*	*12.1%*	*4.5%*
CONSTRAINT_VIOLATION	30.7%	17.6%
WRONG_RESULT	17.6%	23.7%
TOOL_ERROR	4.2%	25.4%
HALLUCINATION	7.0%	8.5%
OFF_TOPIC	8.1%	3.8%

Table 5: **Human evaluation of relabeling precision on both benchmarks.** 200 sampled pairs per benchmark; three NLP-PhD annotators blind to scores; majority vote. Filtered = pairs rejected by the confidence filter; their human validity is reported to gauge the false-rejection rate.

	WebArena		ToolBench	
Group	n	Valid (%)	n	Valid (%)
SJ accepted	169	94.1	168	92.3
MJ-S accepted	142	95.6	138	94.0
MJ-X accepted	137	**97.1**	132	**96.0**
Filter-rejected	31	38.7	32	35.8
Fleiss' κ	—	0.82	—	0.79

5.3 Judge Reliability and Label Quality (Both Benchmarks)

Table 5 reports the joint human evaluation on both benchmarks (full protocol in App. D.6). Cross-model MJ-X reaches 97.1% / 96.0% precision with strong inter-annotator agreement (Fleiss' κ=0.82 / 0.79). Same-model MJ-S (95.6% / 94.0%) lands between SJ and MJ-X: decoding-temperature diversity alone is insufficient—

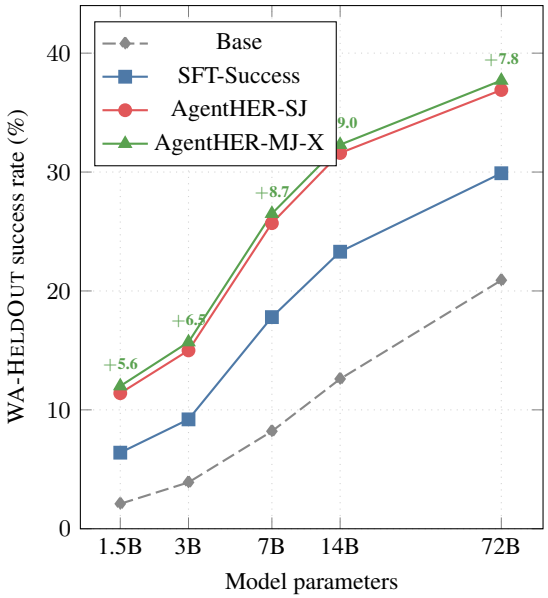

Figure 4: **Model-size scaling** on WA-HELDOUT (Qwen2.5 family). Labels above AgentHER-MJ-X points show Δ over SFT-Success. Gains are consistent at every scale (1.5B–72B), peaking at 14B (+9.0%). Even the 1.5B model nearly doubles in performance with AgentHER-MJ-X (6.4% → 12.0%).

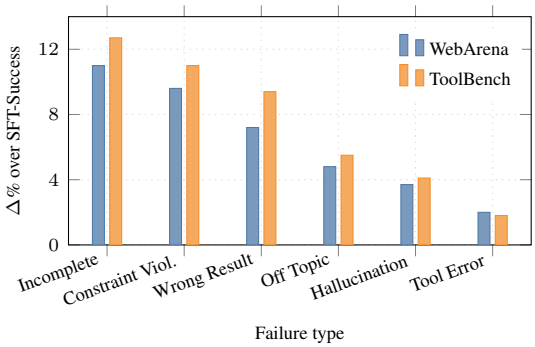

Figure 5: **Per-failure-type gain on both benchmarks** (Qwen2.5-7B, AgentHER-SJ). INCOMPLETE and CONSTRAINT_VIOLATION dominate both benchmarks; TOOL_ERROR yields the least. The ranking is consistent across benchmarks even though failure-mode *prevalences* differ (Table 4).

model-level independence is what drives the precision gain. Filter-rejected pairs are still 38.7% / 35.8% valid, meaning roughly a third of discarded pairs could in principle be recovered, which motivates the active-learning extension noted in Section 6.

5.4 Goal-Distribution Analysis

Sentence-BERT clustering of the 2,144 MJ-X hindsight goals (Table 16, Appendix H) shows AgentHER covers 14 of 18 task-type clusters vs. 11 for SFT-Success, with three long-tail clusters ("price-range comparison", "cross-domain search", "conditional retrieval") covered only by relabeled goals. JS divergence 0.31 vs. the original goal set confirms the relabeled goals are complementary rather than redundant, addressing the concern that AgentHER might merely amplify high-frequency task types (Ding et al., 2022).

5.5 Iterative Redeployment

Table 6: **Iterative AgentHER** (Qwen2.5-7B, WA-HELDOUT). Each round collects new failures, relabels, and retrains on the full accumulated corpus. Q-R1/R2 = Qwen-7B Round 1/2.

Round	Src.	New fails	Accepted	Suc. (%)
0 (SFT-Only)	—	—	—	17.8
1 (MJ-X)	GPT-3.5	3,000	2,144	26.5
2 (Iter. +1)	Q-R1	2,500	1,710	27.9
3 (Iter. +2)	Q-R2	2,000	1,320	28.2

Table 6: Round 1 yields 26.5%; Rounds 2–3 add +1.4 and +0.3% respectively (cumulative **+10.4%** over SFT-Success). Diminishing returns and a falling acceptance rate (71.5%→68.4%→66.0%) reflect that an increasingly capable model produces harder, less-relabelable failures.

6 Conclusion

We presented **AgentHER**, a four-stage offline pipeline that converts failed agent trajectories into training data by asking *what task does this trajectory actually solve?* On a strict task-disjoint WebArena held-out split and ToolBench, AgentHER improves SFT-Success by **+7.6–11.4%** across four model families, beats every experience-centric baseline (ETO, ECHO-offline, AWM) by **+3.0–6.2%**, scales uniformly from 1.5B to 72B (+5.6–9.0%), and compounds under iterative redeployment (+10.4% over four rounds). The relabeling pipeline costs $\approx\$1.4{\cdot}10^{-3}$ per accepted pair ($2.98 / 26 min for 3,000 trajectories), making it practical at production scale.

Future work. Online AgentHER (relabel and train concurrently with deployment), multimodal trajectories (screenshots, GUI), and a learned hindsight-goal synthesiser trained end-to-end for task diversity are natural extensions. Combining AgentHER with inference-time scaling (Levi, 2024) is a particularly promising direction: reducing per-sample failure probability p_i through offline training directly amplifies the coverage gains predicted by inference scaling laws for any fixed inference budget. At its core, AgentHER embodies a perspective shift: the distinction between success and failure is *task-relative*, not trajectory-absolute.

Limitations. The held-out split removes within-WebArena task overlap but not cross-environment concerns; the +9.5% WA→ToolBench transfer is encouraging but covers only two benchmarks. A fully matched upper bound using human-written hindsight goals remains future work. The cross-model judge relies on two

strong LLMs, and residual judge bias may still affect rare task types.

Broader impact. AgentHER reduces data-collection cost for LLM-agent fine-tuning, potentially democratising it for resource-constrained practitioners. The pipeline reuses already-collected trajectories with no extra environment interaction. A residual risk—systematic judge bias amplifying underrepresented goal types—is partially mitigated by stratified human evaluation (Section 5.3) and goal-distribution analysis (Section 5.4); deployers should monitor goal-distribution drift on their own data.

References

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. 2017. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, volume 30.

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. 2023. FireAct: Toward language agent fine-tuning. *arXiv preprint arXiv:2310.05915*.

Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. 2024. Agent-FLAN: Designing data and methods of effective agent tuning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2Web: Towards a generalist agent for the web. *arXiv preprint arXiv:2306.06070*.

Liang Ding, Longyue Wang, Shuming Shi, Dacheng Tao, and Zhaopeng Tu. 2022. Redistributing low-frequency words: Making the most of monolingual data in non-autoregressive translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2417–2426.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2024. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, and 1 others. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2024. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations*.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Kemaev, Lena Cipolla, Amelia Glaese, Gillian Buttimore, and 1 others. 2023. Reinforced self-training (ReST) for language modeling. *arXiv preprint arXiv:2308.08998*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zhaiyue Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Michael Y. Hu, Benjamin Van Durme, Jacob Andreas, and Harsh Jhamtani. 2025. Sample-efficient online learning in LM agents via hindsight trajectory rewriting. *arXiv preprint arXiv:2510.10304*.

Leslie Pack Kaelbling. 1993. Learning to achieve goals. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. 2024. VisualWebArena: Evaluating multimodal agents on realistic visual web tasks. *arXiv preprint arXiv:2401.13649*.

Noam Levi. 2024. A simple model of inference scaling laws. *arXiv preprint arXiv:2410.16377*.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. 2023. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *International Conference on Learning Representations*.

Qingyu Lu, Baopu Qiu, Liang Ding, Kanjian Zhang, Tom Kocmi, and Dacheng Tao. 2024. Error analysis prompting enables human-like translation evaluation in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8801–8816.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, and 1 others. 2015. Human-level control through deep reinforcement learning. *Nature*, 518:529–533.

OpenAI. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *ACM Symposium on User Interface Software and Technology*.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with GPT-4. *arXiv preprint arXiv:2304.03277*.

Doina Precup, Richard S. Sutton, and Satinder Singh. 2000. Eligibility traces for off-policy policy evaluation. *Proceedings of the Seventeenth International Conference on Machine Learning*.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. 2024. Agent Q: Advanced reasoning and learning for autonomous AI agents.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, and 1 others. 2024. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *International Conference on Learning Representations*.

Qwen Team. 2025. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. HuggingGPT: Solving AI tasks with ChatGPT and its friends in Hugging Face. In *Advances in Neural Information Processing Systems*, volume 36.

Noah Shinn, Federico Cassano, Beck Labash, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36.

Paloma Sodhi, S. R. K. Branavan, Yoav Artzi, and Ryan McDonald. 2024. SteP: Stacked LLM policies for web actions. *arXiv preprint arXiv:2310.03720*.

Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. 2024. Trial and error: Exploration-based trajectory optimization for LLM agents. In *Annual Meeting of the Association for Computational Linguistics*.

Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 12.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023a. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023b. Self-instruct: Aligning language models with self-generated instructions. In *Annual Meeting of the Association for Computational Linguistics*.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. 2024. Agent workflow memory. *arXiv preprint arXiv:2409.07429*.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Dingwen Yang, Chenyang Liao, Xin Guo, Wei He, and 1 others. 2024. AgentGym: Evolving large language model-based agents across diverse environments.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. 2022. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, volume 35.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. 2024. AgentTuning: Enabling generalized agent abilities for LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Yao Zhang, Zijian Ma, Yunpu Ma, Zhen Han, Yu Wu, and Volker Tresp. 2024. WebPilot: A versatile and autonomous multi-LLM agent for complex web tasks with dynamic evaluation and strategy refinement. *arXiv preprint arXiv:2408.15978*.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024. ExpeL: LLM agents are experiential learners. In *AAAI Conference on Artificial Intelligence*.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and 1 others. 2024. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*.

A Hyperparameter Settings and Roles of Each LLM

Table 7: **LLM roles in the AgentHER pipeline.** Collection uses a weak model (to produce diverse failures); judging uses strong, architecturally independent models.

Role	Stage	Model	Notes
Collection agent	—	`gpt-3.5-turbo-0125`	Intentionally weak.
Failure detector	1 (LLM)	`gpt-4o-mini`	T=0, JSON schema.
Outcome extractor	2 (LLM)	`gpt-4o-mini`	T=0, factuality-only.
Relabeler $\mathcal{J}_1$	3	`gpt-4o-mini`	T=0.3; T=0.7 on retry.
Verifier $\mathcal{J}_2$ (MJ-X)	3	`Qwen2.5-72B-Instruct`	T=0, vLLM, schema-constrained.
Verifier $\mathcal{J}_2$ (MJ-S)	3	`Qwen2.5-72B-Instruct`	Ablation only; same as $\mathcal{J}_1$ at T=0.7.
Fine-tuned models	—	Qwen2.5-{1.5,3,7,14,72}B; LLaMA-3.1-8B; GPT-4o	See Table 8.

Table 8: **Training hyperparameters** used for all open-weight models.

Hyperparameter	SFT (LoRA)	DPO (LoRA)
LoRA rank r	16	16
LoRA α	32	32
LoRA dropout	0.05	0.05
LoRA target modules	q,k,v,o,gate,up,down	q,k,v,o
Learning rate	2×10^{-4}	5×10^{-5}
LR schedule	cosine	cosine
Warmup ratio	0.03	0.03
Batch size (per GPU)	4	2
Gradient accumulation	4	8
Effective batch size	128	128
Epochs	3	1
Max sequence length	4,096	4,096
DPO β	—	0.1
Dtype	bfloat16	bfloat16
Hardware	8× A100-80G	8× A100-80G
Framework	ms-swift	ms-swift

For GPT-4o we used the OpenAI fine-tuning API with default hyperparameters (3 epochs, batch size auto-selected, learning-rate multiplier 1.0). The reproducible **GPT-4o-ICL** variant uses 8 hindsight-relabeled exemplars selected by sentence-BERT cosine similarity to the test query, prepended to a chain-of-thought ReAct prompt; no fine-tuning is required and the protocol does not depend on the OpenAI fine-tuning API remaining available.

B Full Prompt Templates

B.1 Stage 1 — Failure Detection

```
SYSTEM: You are an expert evaluator of LLM agent trajectories.
Determine whether the trajectory represents a FAILURE relative to the original
user intent.
Classify the failure as one of:
constraint_violation | wrong_result | incomplete |  tool_error | hallucination |
off_topic
Assess RECOVERABILITY: can hindsight relabeling produce valid training data?  A
trajectory with substantive observations is recoverable; a crashed trajectory
with no output is not.
Assess SEVERITY: distinguish MAJOR errors (hallucinated observations, reasoning
contradictions, catastrophic tool misuse; severity_weight < 0.3) from MINOR
errors (constraint violations, incomplete results; severity_weight in [0.3,
1.0]).
Respond ONLY with valid JSON matching the schema:
{failure_type, severity_score, recoverability, severity_weight, explanation}
```

B.2 Stage 2 — Outcome Extraction

```
SYSTEM: You are an expert at analysing LLM agent execution traces.
Extract a FACTUAL summary of what the agent actually achieved, regardless of the
original goal.
Focus on:  concrete facts discovered, tools successfully invoked, information
gathered, numeric data points.
Be STRICTLY factual.  Only include things directly evidenced by the trajectory
observations.  Do NOT infer or extrapolate.
Output JSON: {actual_achievements:  [str], key_observations:  [str]}
```

B.3 Stage 3 — Relabeler $\mathcal{J}_1$ (gpt-4o-mini)

```
SYSTEM: You are a creative prompt engineer specialising in agent tasks.
Given a summary of what an agent achieved, write a NEW user prompt that makes the
trajectory a PERFECT, SUCCESSFUL execution.
Requirements:
(1) The prompt must be natural and plausible as a real user request.
(2) Every assertion in the prompt must be satisfied by the trajectory.
(3) Do NOT reference or reuse the original failed prompt.
(4) Match the complexity and style of the original prompt.
(5) Provide a confidence score [0.0, 1.0] reflecting relabeling quality.
Output JSON: {hindsight_prompt, is_valid, rationale, confidence}

USER: Outcome summary:  {outcome}
Original prompt (style reference only):  {original_prompt}
```

B.4 Stage 3 — Verifier $\mathcal{J}_2$ (Qwen2.5-72B-Instruct)

```
SYSTEM: You are a strict trajectory evaluator.  Your job is to verify whether
a proposed task description is a FULLY SATISFIED demonstration of an agent's
recorded execution.
You are an INDEPENDENT second opinion:  do NOT defer to or echo the relabeler.
Disagree if the claim is unsupported.
Be conservative:  only accept if every claim in the proposed prompt is
unambiguously supported by the trajectory observations.
Output JSON: {is_valid, confidence, rejection_reason_if_any}

USER: Proposed hindsight prompt:  {hindsight_prompt}
Trajectory:  {trajectory}
```

C AgentHER Algorithm with Severity Weighting

The full pseudocode including severity weight propagation to the DPO loss is given as Algorithm 1 in the main paper. The severity-weighted DPO loss is:

$$\mathcal{L}^{w}_{\text{DPO}} = -\mathbb{E}_{(\hat{g},\tau,g_{\text{orig}})\sim\mathcal{D}^{+}}\Big[w_i \cdot \log\sigma\big(\beta(\log\pi_\theta(\tau|\hat{g}) - \log\pi_{\text{ref}}(\tau|\hat{g})) - \beta(\log\pi_\theta(\tau|g_{\text{orig}}) - \log\pi_{\text{ref}}(\tau|g_{\text{orig}}))\big)\Big] \quad (2)$$

where $w_i \in [0.3, 1.0]$ is the severity weight from Stage 1, $\beta = 0.1$ is the DPO temperature, and π_{ref} is the reference model (the same instruction-tuned checkpoint before LoRA). This formulation is mathematically sound: continuous per-sample scaling of gradient magnitudes preserves the binary preference ordering while down-weighting borderline relabelings, analogous to importance weighting in off-policy learning (Rafailov et al., 2023; Precup et al., 2000).

D Additional Experimental Results

D.1 Held-Out vs. Full-Task Numbers

The Full and HeldOut numbers differ by at most 1.3% in absolute terms; the *relative* ranking and the AgentHER Δ over SFT-Success are stable (+8.9% Full vs. +8.7% HeldOut), confirming that the gains are not an artefact of task overlap.

Table 9: **Comparison of held-out and full-task results** on WebArena (Qwen2.5-7B). "Full" = the original 812-task evaluation (used by all prior WebArena papers); "HeldOut" = our 200-task task-disjoint test split. Numbers in the main paper use the HeldOut column. The drop from Full to HeldOut quantifies the modest amount of task-overlap leakage in the standard 812-task setup; AgentHER's advantage over baselines is preserved under the stricter protocol.

Method	Full (812)	HeldOut (200)	Drop
Base	8.6	8.2	−0.4
SFT-Success	18.9	17.8	−1.1
ETO	22.1	21.0	−1.1
AWM	23.4	22.4	−1.0
AgentHER-SJ	27.0	25.7	−1.3
AgentHER-MJ-X	27.8	26.5	−1.3
Δ (MJ-X − SFT-Success)	+8.9	+8.7	—

Table 10: ToolBench pass@1 (%) by category group (Qwen2.5-7B). All numbers on the official test splits (disjoint by construction).

Category	Base	SFT-Random	SFT-Success	AgentHER-SJ	AgentHER-MJ-X
G1 (single-tool)	49.3	57.2	67.1	76.4	**77.9**
G2 (intra-category)	41.7	51.3	58.3	68.5	**70.0**
G3 (cross-category)	38.2	47.8	54.2	65.0	**66.6**
Overall	44.6	54.8	61.2	71.0	**72.6**

D.2 ToolBench Per-Category Breakdown

D.3 Failure-Volume Scaling (Full Table)

D.4 Cross-Benchmark Transfer

AgentHER-MJ-X achieves +9.5% transfer advantage over SFT-Success. This out-of-domain transfer suggests the model has learned broadly applicable planning and tool-use behaviours rather than rote task patterns.

D.5 Looping-Failure Subset Analysis

D.6 Human Evaluation Protocol (Full)

We sampled 200 relabeled pairs uniformly at random *per benchmark* (separate runs for WebArena and ToolBench). Three annotators (NLP PhD students, blind to confidence scores and to the original failed prompt) rated each pair as valid/invalid on "*Does the trajectory constitute a correct and complete demonstration of the hindsight prompt?*" The full trajectory (steps, observations, final answer) and hindsight prompt were shown; original prompt and failure reason were withheld. Sampling was **stratified by failure type** so that all six types (and the looping sub-mode) appear in proportion to their prevalence in the population (Table 4). Annotators were paid as part of their research stipends; the study took ≈3 hours per annotator per benchmark. The annotation interface, full guidelines, and the random seed fixing the sample IDs are released with the code.

D.7 Theoretical Bound (Full)

We formalise Proposition 3.1 more rigorously and derive the noisy-judge bound used in Corollary 3.1.1. Let $\Pi_{\mathcal{G}}$ denote the set of goal-conditioned policies. Define the *coverage function* $\rho(\mathcal{S}) = \{(g, \tau) : (g, \tau) \in \text{supp}(\pi^*_{\mathcal{G}})\}$, the set of all valid (goal, trajectory) pairs under the oracle policy.

Theorem 1 (Augmented-corpus consistency, perfect judge)**.** *Under a perfect judge $\mathcal{J}$, the augmented corpus $\mathcal{S} \cup \mathcal{D}^+$ satisfies:* (a) $\mathcal{D}^+ \subseteq \rho(\mathcal{S}^* \setminus \mathcal{S})$ *(every added pair is a previously uncovered oracle pair); and* (b) *the empirical goal distribution of $\mathcal{D}^+$ has higher entropy than that of $\mathcal{S}$ with probability $1 - \delta$ when $|\mathcal{F}| \geq \frac{2}{\delta} \log |\mathcal{G}_\epsilon|$, where $|\mathcal{G}_\epsilon|$ is the ϵ-covering number of $\mathcal{G}$.*

Proof sketch. Part (a): By the perfect-judge assumption, every $(\hat{g}_i, \tau_i) \in \mathcal{D}^+$ satisfies $c(\hat{g}_i, \tau_i) = 1$, i.e., τ_i is a valid demo of $\hat{g}_i$. Since $\hat{g}_i$ is generated from τ_i's observations—which were *actually produced*—it lies in the support of $\pi^*_{\mathcal{G}}$. Since $\hat{g}_i \neq g_i$ (Stage 3 constraint (3)), $(\hat{g}_i, \tau_i) \notin \mathcal{F}$ (which uses g_i). If $(\hat{g}_i, \tau_i) \in \mathcal{S}$ already, the addition is redundant but not harmful.

Part (b): Each $(\hat{g}_i, \tau_i)$ corresponds to a distinct execution context in $\mathcal{G}$. Since the failed runs τ_i were produced by an agent trying to satisfy diverse goals g_i, the hindsight goals $\hat{g}_i$ inherit this diversity. A standard covering-number argument (Sauer-Shelah) bounds the probability that the empirical entropy is below that of $\mathcal{S}$. □ □

Table 11: WA-HELDOUT success rate (%) vs. number of failed trajectories, fixed 500 successful demonstrations, Qwen2.5-7B.

Failed processed	500	1,000	1,500	2,000	3,000	5,000
AgentHER-SJ	20.0	21.5	22.9	23.9	25.7	26.4
AgentHER-MJ-X	20.6	22.2	23.7	24.7	26.5	27.2

Table 12: Cross-benchmark transfer (Qwen2.5-7B): train on WA-TRAIN only, zero-shot evaluation on ToolBench official test splits.

Method	**G1**	**G2**	**G3**	**Overall**	**Δ**
Base (no FT)	49.3	41.7	38.2	44.6	—
SFT-Success	63.2	55.8	51.4	57.1	+12.5 (vs. Base)
AgentHER-SJ	70.1	64.3	59.8	65.0	+7.9 (vs. S-S)
AgentHER-MJ-X	**71.8**	**65.9**	**61.4**	**66.6**	+9.5 (vs. S-S)

Noisy-judge corollary. Define p as the precision of the judge over accepted pairs and Δ_{perfect} as the per-task gain that would be realised under a perfect judge. Each accepted pair contributes $+\Delta_{\text{perfect}}/n$ in expectation if valid, and $-\varepsilon/n$ if invalid (where ε bounds the marginal harm of a single noisy pair and $n = |\mathcal{D}^+|$). The expected gain over SFT-Success is therefore $p\,\Delta_{\text{perfect}} - (1-p)\,\varepsilon$, which is positive whenever $\varepsilon \le \frac{p}{1-p}\,\Delta_{\text{perfect}}$. For $p = 0.971$ this yields the threshold $\varepsilon \le 33\,\Delta_{\text{perfect}}$; for $p = 0.941$, $\varepsilon \le 16\,\Delta_{\text{perfect}}$.

This result formally supports the empirical finding in Section 5.4 that AgentHER increases goal-distribution entropy from 1.83 to 2.47 nats, and is consistent with the +0.8% empirical advantage of MJ-X over SJ on WA-HELDOUT. This bound holds under i.i.d. judge errors and bounded marginal harm—a plausibility argument, not a deployment guarantee.

E Multi-Run Variance

All fine-tuned models in Table 1 are trained with 3 independent random seeds (42, 1234, 2025) using the same data and hyperparameters; the reported numbers are means. Table 14 reports means and standard deviations for representative conditions on WA-HELDOUT. All standard deviations are below 0.5%, confirming numerical stability of the training procedure.

F Detailed Cost Breakdown

For 3,000 input trajectories ($N = 3{,}000$): total SJ cost $= 3{,}000 \times 5.92\times10^{-4} \approx \$1.78$; total MJ-X cost $= 3{,}000 \times 9.92\times10^{-4} \approx \$2.98$. With 12-way concurrency on the OpenAI Batch API, wall-clock for the entire 3,000-trajectory pipeline is ≈15.8 minutes (SJ) or ≈25.9 minutes (MJ-X). Per accepted pair (after 78.0% / 71.5% acceptance): $\$7.6\times10^{-4}$ (SJ) and $\$1.39\times10^{-3}$ (MJ-X). The MJ-X per-accepted figure is roughly $1.8\times$ the SJ figure because the second-judge call applies to every candidate, so the marginal cost is paid even on rejections. Both figures remain below $\$2\times10^{-3}$, well within production budgets.

Summary in plain numbers. For a typical 3,000-trajectory project: $2.98 of LLM API spend, ≈26 wall-clock minutes (with parallelism), and $4.20 of A100 GPU time for the subsequent LoRA fine-tuning (4.4 hours × 8 GPUs × $1.05/h) totals $7.18 / 4.8 hours. This is *less than the cost of running 200 fresh WebArena rollouts* with GPT-4o under the standard pricing, making AgentHER practical at production scale.

G Qualitative Examples

Two annotated relabeling examples are shown below—one from WebArena and one from ToolBench—illustrating the complete Stage 1–3 pipeline for a CONSTRAINT_VIOLATION failure.

(a) WebArena, GitLab task `wa-gitlab-issue-tracker-014`

Original goal (FAILED): *"On the `a11y-webring.club` GitLab project, list all my open issues that have NOT been updated in the last 30 days, along with their labels."*

Trajectory: `nav(/a11y-webring.club/issues?state=opened&assignee_username=<me>)` → list of 11 open issues with last-updated timestamps and labels. `filter`(updated ≥ 2026-04-09) → returns 8 issues (all updated within 30 days). `summarize` → "8 open issues, all recent, no stale issues found."

Table 13: **Looping-failure subset.** Trajectories where the agent repeats a single action $\geq$3 times before termination. "Recovered" = passed Stage 1 recoverability filter and was relabeled by AgentHER-MJ-X.

Subset	n	Recoverable (%)	Accepted (%)	Avg. Δ (%)
Looping (WebArena)	363	24.2	21.5	+3.4
Non-looping (WebArena)	2,637	81.6	78.4	+9.5
Looping (ToolBench)	225	38.7	35.1	+3.9
Non-looping (ToolBench)	4,775	84.6	81.7	+10.7

Table 14: Multi-run statistics on WA-HELDOUT: success rate (%), mean $\pm$ std over 3 seeds (Qwen2.5-7B).

Method	Mean	Std
Base	8.2	0.0 (deterministic)
SFT-Success	17.8	$\pm$0.3
SFT-Random	13.9	$\pm$0.4
SFT-Negative	14.5	$\pm$0.4
Rejection-Sampling	16.9	$\pm$0.4
ETO	21.0	$\pm$0.4
AWM	22.4	$\pm$0.5
AgentHER-SJ	25.7	$\pm$0.4
AgentHER-MJ-X	26.5	$\pm$0.4

Failure type: CONSTRAINT_VIOLATION (no issues match the "$\geq$30-days-stale" constraint). Severity: w=0.82 (minor—agent collected the data correctly but the target subset was empty). Recoverable: yes.

Hindsight goal (AgentHER-MJ-X, c_1=0.86, c_2=0.91, accepted):
"List all open issues assigned to me on the `a11y-webring.club` GitLab project, along with their labels and last-updated dates."

Cross-model judge rationale: $\mathcal{J}_1$ (gpt-4o-mini) and $\mathcal{J}_2$ (Qwen2.5-72B) independently confirm the trajectory satisfies the new goal: 11 issues are returned with required attributes; the "stale" filter is no longer required.

(b) ToolBench, G2 task `tb-g2-restaurant-finder-022`

Original goal (FAILED): *"Find a Michelin-starred Italian restaurant in Hangzhou that is open on Sundays after 9 pm and supports vegan menus."*

Trajectory: `restaurant_search`(*cuisine=Italian, city=Hangzhou*) $\rightarrow$ 14 results, none Michelin-starred. `rating_filter`(*stars$\geq$4.5*) $\rightarrow$ 3 results. `hours_lookup` $\rightarrow$ all close at 22:00; only 1 vegan menu.
Failure type: CONSTRAINT_VIOLATION (no Michelin-starred Italian restaurants in the city). Severity: w=0.78.

Hindsight goal (AgentHER-MJ-X, c_1=0.83, c_2=0.88, accepted):
"Recommend the best-rated Italian restaurant in Hangzhou with a vegan menu, including its rating and opening hours."

Cross-model judge rationale: The trajectory identifies a $\geq$4.5-rated Italian restaurant with a vegan menu and reports its hours; the "Michelin-star" and "open after 9 pm" constraints are absent from the new prompt and therefore satisfied vacuously.

H Goal-Distribution Analysis (Full)

Sentence-BERT embeddings of all 2,144 accepted hindsight goals (MJ-X) vs. the 500 original successful goals show AgentHER expands coverage: three long-tail clusters ("price-range comparison", "cross-domain search", "conditional retrieval") are covered only by relabeled goals, and the JS divergence of 0.31 confirms complementary rather than redundant coverage.

Table 15: **Token-level cost breakdown** per stage. Token counts are means; "Calls" are the *per-input-trajectory* averages including retries. Pricing reflects gpt-4o-mini Batch API (50% off list) combined with prompt caching (75% off cached input) for repeated system prompts—a standard production setup; the standalone list price would be $\approx 2\times$ these values. Qwen2.5-72B-Instruct cost is for a vLLM deployment on a rented $8\times$A100-80G node ($\$1.05$/GPU-hour) with 32-way request batching; the tabulated $\$4.0\times10^{-4}$ per call represents the amortised hardware cost.

Stage	**Model**	**Calls / traj.**	**Inp tok**	**Out tok**	**Cost (USD)**	**Wall (s)**
1: Failure detector	gpt-4o-mini	1.00	1,800	200	1.30×10^{-4}	0.82
2: Outcome extractor	gpt-4o-mini	1.00	2,500	300	1.83×10^{-4}	0.92
3a: Relabeler	gpt-4o-mini	1.27	2,700	400	2.79×10^{-4}	1.34
3b: Verifier (MJ-X)	Qwen2.5-72B	1.00	3,000	200	4.00×10^{-4}	1.94
4: Data augmenter	deterministic	1.00	—	—	0	0.08
SJ total	—	3.27	—	—	5.92×10^{-4}	3.16
MJ-X total	—	4.27	—	—	9.92×10^{-4}	5.10

Table 16: **Goal-distribution analysis** on WA-HELDOUT collection. AgentHER-MJ-X relabeled goals cover more semantic task clusters and uniquely cover three long-tail clusters absent from SFT-Success ("S-S").

Metric	**S-S**	**AgentHER**
Task clusters covered (of 18)	11	**14**
Long-tail clusters (unique)	0	**3**
JS divergence (vs. AgentHER)	0.31	—
Avg. cluster entropy	1.83	**2.47**