WORKSYNTH: OFF-POLICY SYNTHETIC DATA FOR PROFESSIONAL ARTIFACT AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Professional work benchmarks such as GDPval have begun to evaluate language agents not by isolated answers, but by their ability to produce complete, usable artifacts such as reports, spreadsheets, slide decks, analyses, and decision documents. This shift exposes a gap in current post-training practice. While recent reinforcement-learning methods have achieved strong gains on tasks with compact and verifiable outcomes, professional artifact generation is characterized by underspecified goals, heterogeneous formats, domain-specific conventions, and evaluation criteria that are only partially reducible to pass–fail rewards. We argue that part of this limitation reflects a distributional mismatch: general instruction-tuning data rarely teaches models the forms of situated work required to transform complex requests and supporting materials into polished deliverables. This paper studies whether such capabilities can be improved through off-policy post-training rather than expensive online agentic reinforcement learning. We construct synthetic professional work supervision that varies along three dimensions: task formulation, artifact structure, and revision-oriented feedback. Across GDPval-style evaluations, we examine whether models benefit more from additional final-output demonstrations, from critique-and-revision supervision, or from data targeting specific artifact conventions and professional genres. Our results show that synthetic work-oriented supervision improves instruction adherence, structural completeness, and deliverable quality, with the largest gains arising when synthetic data represents not only the desired final artifact but also the evaluative criteria that distinguish acceptable from incomplete work. These findings suggest that professional artifact agents may be limited less by a lack of generic reasoning ability than by insufficient exposure to the conventions, constraints, and revision practices of real-world knowledge work.

1 INTRODUCTION

Language-model agents are increasingly judged by the products they deliver, rather than by isolated answers. GDPval asks systems to transform occupational requests and reference files into usable documents, spreadsheets, presentations, diagrams, and multimedia artifacts (Patwardhan et al., 2026). Software-agent benchmarks likewise require repository patches that survive execution, rather than plausible textual explanations (Jimenez et al., 2024; Miserendino et al., 2025), while desktop benchmarks test workflows spanning files and applications (Xie et al., 2024). Across these settings, success depends jointly on factual correctness, instruction adherence, artifact structure, domain conventions, and evidence that the agent checked and repaired its work.

This shift exposes a mismatch between the target behavior and common post-training data. General instruction corpora provide broad language and reasoning coverage, but rarely demonstrate how a situated request, supporting files, professional conventions, and evaluation criteria become a finished work product. Online agentic reinforcement learning can generate such experience, but repeatedly executing long trajectories in stateful sandboxes makes optimization expensive and sensitive to sparse rewards, evaluator failures, and interaction horizons. This motivates our central question: ***Can professional-artifact competence be learned from reusable off-policy supervision, and what makes a synthetic work trajectory worth training on?***

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

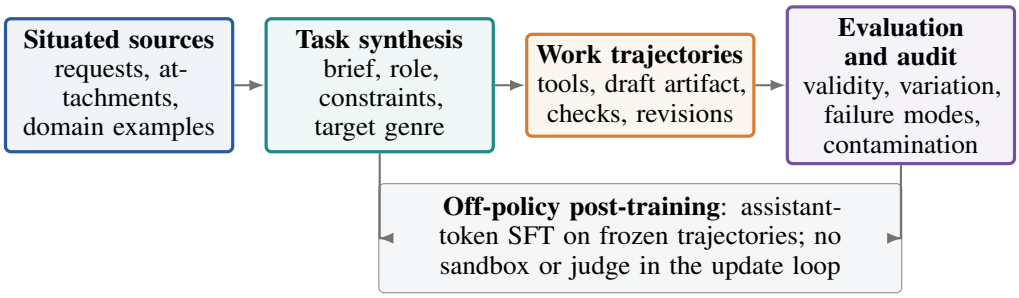

Figure 1: WORKSYNTH turns situated work into frozen, auditable supervision. The professional-work branch creates tasks, reference artifacts, rubrics, and multi-model rollouts; a complementary software-artifact branch provides a controlled frozen-trajectory training study. Online execution is used for data creation and evaluation, not for gradient updates.

We study this question with WORKSYNTH, a pipeline for constructing and auditing frozen work trajectories before optimization. WORKSYNTH varies supervision along three axes: task formulation from situated source material, the genre and structural contract of the target artifact, and the presence of evaluation-triggered checks and revisions. Explicit criteria provide task validation, failure diagnosis, and candidate selection metadata. The student is then trained with assistant-token supervised fine-tuning: sandbox execution may be used to create and evaluate data, but no environment or judge is invoked in the gradient-update loop.

Empirical scope. The completed experiments do not yet implement a matched off-policy GDPval-style ablation of final-output, critique-and-revision, and genre-targeted supervision. The controlled off-policy comparison concerns software artifacts; the professional-work evidence consists of construction and scorer diagnostics plus an online mixture pilot. We therefore treat the relative benefit of the three professional supervision forms as a hypothesis rather than an isolated causal finding.

Our evidence comprises two deliberately separated branches because no available setting provides both the breadth of professional office artifacts and a large, executable controlled training comparison. The *professional-work branch* synthesizes heterogeneous tasks, reference deliverables, rubrics, and multi-policy rollouts. It tests whether the resulting tasks and evaluators expose meaningful quality differences, and provides a preliminary held-out study of artifact-specific exposure. The *software-artifact branch* supplies the controlled frozen-trajectory experiment: the same 9B student is trained on a historical rule-selected subset, an equal-size random subset, or the complete candidate pool, and all variants are evaluated on the 500 tasks of SWE-bench Verified. Software patches are an adjacent executable artifact domain, not a substitute for a controlled GDPval-scale training study; the two branches therefore answer complementary questions.

The results support three evidence-bounded conclusions. First, frozen-trajectory SFT is effective in the executable-artifact control: resolve rate rises from 1.6% to 9.8% with 6,500 randomly sampled traces and 10.8% with all 16,728 candidates. A historical rule-selected subset reaches 14.4%, although a post-hoc rule audit and an unmatched 27B reversal make this selection result exploratory rather than a validated quality rule. Second, in separately collected professional-work cohorts, the observed fraction of Qwen-family tasks with high within-task reward variation changes from 20% to 40%, while a task-level diagnostic keep rate changes from 38% to 54%. These are descriptive pipeline diagnostics, not a paired effect of rubric redesign. Third, a 43-task online pilot reveals a distributional gap: skill-only training reduces GDPval-style score from 0.160 to 0.056, whereas mixing artifact tasks reaches 0.237. This pilot motivates artifact-specific supervision but is not itself an off-policy ablation. The current experiments also do not isolate revision tokens while holding tasks and final artifacts fixed.

Our contributions are:

- **A formulation and pipeline for work-oriented supervision.** We represent situated task context, artifact conventions, criterion-level evidence, and feedback-triggered repair as components of the post-training distribution, and instantiate this formulation across heterogeneous professional artifacts.

- **A controlled frozen-trajectory training study.** On executable software artifacts, random and full replay substantially improve a 9B student; we also report both the gain and the robustness failure of a historical rule-selected subset.
- **Diagnostics for professional-artifact data quality.** We quantify evaluator brittleness, sparse reward, and missing artifact exposure, while distinguishing descriptive diagnostics from the critique-only and same-checkpoint GDPval ablations that remain future work.

2 Related Work

Benchmarks for situated agent work. SWE-bench introduced repository-level issue resolution with executable tests (Jimenez et al., 2024); SWE-Gym subsequently studied training agents and verifiers in related environments (Pan et al., 2025). OpenHands provides a general sandboxed software-agent platform (Wang et al., 2025). OSWorld evaluates open-ended computer use across real applications (Xie et al., 2024), while GDPval moves beyond interaction success to occupational work products with reference material, heterogeneous formats, and expert-authored rubrics (Patwardhan et al., 2026). Format-specific studies include spreadsheet manipulation (Ma et al., 2024), presentation generation and evaluation (Zheng et al., 2025), and rubric-based research replication (Starace et al., 2025). WorkSynth focuses on post-training for the artifact-centric regime, using software artifacts as a controlled adjacent domain.

Trajectory and tool-use post-training. ReAct interleaves reasoning and environment actions (Yao et al., 2023). AgentTuning filters interaction trajectories and mixes them with general instructions to elicit agent ability (Zeng et al., 2024); Agent-FLAN decomposes agent data into format following and reasoning components (Chen et al., 2024). These works establish that trajectory supervision can activate tool use, but professional artifacts add a target object whose layout, internal consistency, and genre conventions must also be learned. Our random and full arms test frozen-trajectory learning, while the audited rule-selected arm probes—but does not validate—data selection.

Synthetic data and revision. Self-Instruct bootstraps instructions from model generations (Wang et al., 2023), and AgentInstruct uses agentic flows to generate large post-training corpora from raw sources (Mitra et al., 2024). Synthetic data quality and diversity remain central concerns. Self-Refine (Madaan et al., 2023) and Reflexion (Shinn et al., 2023) use natural-language feedback without parameter updates. SCoRe shows that offline SFT on model-generated correction traces can be insufficient for test-time self-correction because of policy-distribution mismatch and behavior collapse (Kumar et al., 2025). WorkSynth asks a different question—whether frozen work trajectories improve final artifact production—and retains checks, failures, and repairs when they occur in logged interactions, while criteria and post-hoc judgments remain audit metadata unless explicitly observed by the agent.

Offline and online alignment. Instruction tuning and RL from human feedback combine demonstrations, preferences, and online policy optimization (Ouyang et al., 2022). Direct Preference Optimization provides a simpler offline objective for preference data (Rafailov et al., 2023), whereas GRPO removes a learned critic by normalizing rewards within prompt groups (Shao et al., 2024). Our central method is deliberately simpler—assistant-token supervised fine-tuning (SFT) on frozen trajectories—so gains can be attributed to the supervision distribution rather than an online optimizer. We nevertheless report online GRPO controls to quantify both its attainable improvement and its operational sensitivity.

3 Professional Artifact Supervision

3.1 Data records and terminology

We represent a professional task as $x = (r, b, A, g)$, where r is the requested role, b is the work brief, A is a set of supporting attachments, and g is the required artifact genre. A behavior policy μ produces an interaction

$$\tau = (h_0, a_1, o_1, \ldots, a_T, o_T, y), \tag{1}$$

where h_0 contains the task and tool definitions, a_t is an assistant message or tool call, o_t is the resulting observation, and y is the external artifact bundle. For professional documents, y may contain PDF, DOCX, XLSX, or PPTX files; for software tasks it is a repository patch.

A task-specific evaluator E_C, parameterized by criteria $C = (c_1, \ldots, c_K)$, returns criterion outcomes $e = E_C(x, y)$ and an aggregate score $s \in [0, 1]$. We store $d = (x, \tau, y, C, e, s)$. The serialized task and interaction provide model inputs and targets, whereas artifacts, criteria, and scores are primarily validation and selection metadata. Post-hoc judge rationales are not appended to the training sequence unless the corresponding feedback already occurred as a tool observation in the recorded interaction.

We use *off-policy* only in the data-collection sense: trajectories are generated by fixed behavior policies, frozen before optimization, and may come from policies other than the student. Training is masked behavior cloning on logged trajectories. It is not off-policy RL: we use no importance ratios, critic, policy-gradient estimator, or environment interaction during parameter updates.

3.2 Professional-work construction

Task formulation. Given source material, a teacher constructs a situated brief specifying a role, operational context, attachments, required deliverables, and constraints. The target genre is controlled explicitly, while wording is varied to avoid reproducing the source task.

Reference construction and preflight. A tool-using teacher attempts the task in a sandbox. The resulting reference artifact is checked for existence, parseability, and format-specific internal structure. The reference supports evaluator construction and validation; it is not exposed to rollout policies at evaluation time.

Criterion construction. Criteria cover structural validity, source-grounded facts, semantic quality, and penalties. Deterministic checks are used when correctness is invariant to implementation choices, while an LLM judge handles semantic requirements. The central constraint is solution invariance: a criterion specifies what the deliverable must satisfy, not an incidental coordinate, filename, or layout chosen by the reference. Thus a required assumptions table may be checked by its contents, but not by requiring its label to occupy cell A7 unless the brief itself specifies A7.

Repeated rollout collection. Multiple behavior-policy families make repeated attempts using shell and file tools. Each record preserves the complete interaction and produced files. When a rollout encounters a parser error, failed check, or test failure and subsequently repairs it, both the triggering observation and corrective action remain in the trace. The present experiments retain these full traces; they do not yet provide a matched causal comparison against a final-output-only projection.

Validity and training-signal audit. Reference preflight records whether the artifact and evaluator can be executed. In the recorded construction run, rollout launch was not strictly gated on this report; we audit that implementation gap explicitly rather than treating every report as a pass. For diagnostic analysis, given N repeated rollouts from policy μ, we compute

$$\widehat{\sigma}_\mu(x) = \left[\frac{1}{N-1} \sum_{i=1}^{N} (s_i - \overline{s})^2 \right]^{1/2}. \tag{2}$$

We report $\widehat{\sigma} < 0.05$, $0.05 \le \widehat{\sigma} < 0.10$, and $\widehat{\sigma} \ge 0.10$ as low-, intermediate-, and high-variation bins. These are empirical reporting thresholds, not universal conditions for learnability or hard filters in the SFT experiment.

A separate trajectory-aware audit examines all attempts for a task and labels three properties: whether score differences correspond to meaningful behavior differences, whether low-scoring attempts expose an identifiable failure mode, and whether the rubric discriminates consistently. A task is retained only when all three properties pass. Because the decision is produced by an LLM and is applied to a collection of rollouts, we call it a *task-level diagnostic*, not a trajectory-level ground-truth label.

3.3 FROZEN-TRAJECTORY SFT

Let $\xi_i = \mathrm{Serialize}(x_i, \tau_i)$ be the student-template serialization of a frozen record, and let m_{it} equal one only for assistant tokens. Tool observations remain in the conditioning context but receive zero loss. The objective is

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\frac{\sum_i \sum_t m_{it} \log \pi_\theta(\xi_{it} \mid \xi_{i,<t})}{\sum_i \sum_t m_{it}}. \tag{3}$$

Scores and criterion outcomes affect construction or dataset membership but do not weight token loss.

3.4 CONTROLLED SOFTWARE-ARTIFACT VARIANTS

The professional branch establishes the synthesis and evaluator-audit pipeline but does not yet contain a completed same-checkpoint offline SFT matrix. We therefore test frozen-trajectory training in an adjacent executable-artifact setting using software-agent logs. Shared preprocessing removes evaluation holdouts, renders the student template, rejects sequences above 65,536 tokens, validates tool-call arguments, and rejects examples with no assistant loss. This produces a candidate universe $\mathcal{C}$ of 16,728 trajectories. We compare

$$\mathcal{D}_{\mathrm{full}} = \mathcal{C}, \tag{4}$$

$$\mathcal{D}_{\mathrm{rand}} \sim \mathrm{UniformSubset}(\mathcal{C}, 6500), \tag{5}$$

$$\mathcal{D}_{\mathrm{rule}} = \{\tau \in \mathcal{C} : q_{\mathrm{v1}}(\tau) = 1\}, \tag{6}$$

where $|\mathcal{D}_{\mathrm{rule}}| = 6500$ and the random permutation uses NumPy seed 0. The frozen historical rule requires a passing source outcome, reasoning coverage at least 0.64, at least one post-fix verification, zero blind-retry ratio, format-failure rate at most 0.05, and effective-step rate at least 1.0.

We call this arm *Rule-selected (v1)*, not *criteria-selected*. A retrospective audit found that reasoning coverage was directionally confounded with source failure, the format gate was nearly inert, two process metrics were almost redundant, and post-fix verification was weakly associated with outcome. The completed experiment therefore measures the empirical effect of one frozen membership rule; it does not validate those process gates as a general trajectory-quality function.

Rule-selected versus Random holds example count fixed but not tokens: the pools contain 338.0M and 318.8M rendered tokens, respectively, a 6.0% difference. Full contains 819.4M tokens. Because every arm trains for two epochs, Full also receives more optimizer updates; its comparison is equal-epoch rather than equal-compute.

4 EXPERIMENTAL SETUP

4.1 DATASETS

Professional-work synthesis. The pipeline generated 132 task briefs, 92 reference deliverables, 88 rubrics, and 83 verification reports; 48 reports meet the recorded 0.95 integrity threshold. Fifty-seven tasks received multi-policy rollouts, yielding 784 API-complete, scored attempts across Qwen-, GLM-, Kimi-, and DeepSeek-family policies; this count does not imply that an attempt produced a usable artifact. The deliverables include PDF, DOCX, XLSX, PPTX, and mixed file bundles. A task-level trajectory audit retained 29 of 54 inspected tasks. Table 1 reports the inventory.

Off-policy software-artifact control. We begin with 95,828 tool-use trajectories covering 45,170 unique software tasks. We exclude a 499-task source-corpus holdout before training, render each trajectory with the student tokenizer, discard trajectories longer than 65,536 tokens or containing invalid tool-call arguments, and obtain a 16,728-example candidate universe. A separate post-construction audit confirms zero instance-ID overlap against all 500 SWE-bench Verified evaluation tasks. Rule-selected (v1) and Random each contain 6,500 trajectories. Their median lengths differ by 6.1%; the pools contain 338.0M and 318.8M tokens, respectively.

GDPval-style held-out split. We use a fixed seed-42 occupationally stratified split with 177 training and 43 test tasks. Test tasks are excluded from training. We also use an 18-task SkillBench holdout to measure primitive tool-skill retention.

Table 1: Professional-work synthesis inventory. Rollout launch was not a strict subset of integrity passes in the historical run; "retained" is measured on the 54-task audited subset.

Stage	Count	From briefs
Task briefs	132	100%
Reference deliverables	92	70%
Rubrics	88	67%
Verification reports	83	63%
Integrity passes	48	36%
Tasks with new-rubric rollouts	57	43%
Diagnosed and retained	29	22%

4.2 Models and training

The off-policy control fine-tunes the Qwen3.5-9B checkpoint (Qwen Team, 2026). All arms use two epochs, global batch size 128, Adam with peak learning rate 10^{-5} and cosine decay to 10^{-6}, 10% warmup, weight decay 0.1, and a 65,536-token cutoff. Loss is applied only to assistant turns. Rule-selected and Random use identical example counts and schedules; Full has more optimizer updates because it exposes all candidates, so the comparison changes both membership and total optimization.

The GDPval-style control uses a GLM-family 30B mixture-of-experts checkpoint with 3B active parameters and a tool-using agent scaffold. The 30-step pilot compares the base model, skill-only training, and a same-step mixture of skill tasks with artifact tasks. A longer online GRPO run uses eight samples per prompt, 16,384 response tokens, and the same artifact evaluator. We report this run as a reference point, not as part of the off-policy method.

4.3 Evaluation and contamination

For software artifacts, we run all 500 SWE-bench Verified instances in isolated execution environments and report resolved rate. Training and evaluation have zero exact instance-ID overlap. A character 5-gram MinHash screen over 45,170 training issue descriptions and all 500 evaluation statements finds zero pairs with exact Jaccard similarity at or above 0.8. At the repository level, 269 of 500 evaluation instances belong to repositories seen during training. We therefore also report the 231 Django tasks, whose repository is absent from training. Pairwise uncertainty uses instance-level bootstrap confidence intervals and exact McNemar tests.

For professional artifacts, deterministic checks score file validity and exact requirements, while a fixed judge scores semantic criteria. We report mean score, nonzero mean, zero-count, artifact production, and within-task variance. Because a judge error can resemble model failure, missing judge responses are excluded rather than assigned zero.

5 Results

5.1 Frozen trajectories improve the 9B agent; the v1 rule is exploratory

Table 2 gives the main controlled result. Random frozen-trajectory SFT raises resolve rate from 1.6% to 9.8%, and Full reaches 10.8%. Rule-selected (v1) reaches 14.4%, 4.6 points above the equal-size Random arm (paired bootstrap 95% CI $[1.6, 7.6]$; exact McNemar $p = .004$) and 3.6 points above Full (CI $[0.4, 6.8]$; $p = .033$). On the zero-repository-overlap Django subset, Rule-selected reaches 13.0% versus 5.2% for Random ($p < .001$) and 8.7% for Full ($p = .087$).

The base-to-Random gain establishes that logged agent traces can teach environment-facing behavior without executing the environment during optimization. The v1 result further shows that dataset membership matters in this 9B setting, but it does not identify why. The rule includes a source-success gate, is not token matched to Random, and its process metrics fail several post-hoc validity checks. The most defensible explanation is a composition effect, especially the success gate, rather than a causal benefit of the five process thresholds.

Table 2: Frozen-trajectory SFT on SWE-bench Verified. Random and Rule-selected (v1) are matched at 6,500 trajectories. Django is the 231-task zero-repository-overlap subset.

Training data	Traj.	All resolved	All	Django resolved	Django
Base model	0	8 / 500	1.6%	3 / 231	1.3%
Random	6,500	49 / 500	9.8%	12 / 231	5.2%
Full	16,728	54 / 500	10.8%	20 / 231	8.7%
Rule-selected (v1)	6,500	**72 / 500**	**14.4%**	**30 / 231**	**13.0%**

Table 3: Observed rollout diagnostics before and after pipeline/rubric revision on non-matched cohorts. "High variation" means within-task score standard deviation of at least 0.10. Semantic LLM-judge criteria were omitted for throughput.

	High-variation task rate		Mean within-task std.		Task-level audit	
Rubric	Qwen	GLM	Qwen	GLM	Keep rate	Discrimination failure
Original	20% (15/75)	8% (2/26)	0.062	0.035	38% (18/47)	45%
Redesigned	**40% (22/55)**	**22% (9/41)**	**0.107**	**0.063**	**54% (29/54)**	**17%**

This ordering is not robust across training settings. In an unmatched Qwen3.6-27B check, Full resolves 251/500 tasks (50.2%) while Rule-selected (v1) resolves 177/500 (35.4%). Rule-selected uses two epochs (approximately 676M token exposures) and Full one epoch (819M), with no 27B Base or Random control. The comparison does not identify a scale effect; it establishes that the v1 rule is not a universal quality function.

5.2 Professional rubric diagnostics are cohort-dependent

Our first professional-work rubric over-specified a reference artifact. Spreadsheet criteria often fixed content to a particular cell and exact Unicode punctuation, so semantically valid alternative layouts failed. The redesigned rubric prohibits incidental coordinates, normalizes typography, supports filename globs, and reserves semantic judgments for the judge. Table 3 summarizes separately collected cohorts; it is a pipeline diagnostic, not a paired causal estimate.

Across these non-matched cohorts, the observed Qwen high-variation rate is 20% versus 40%, and the GLM rate is 8% versus 22%. The task-level keep rate is 38% versus 54%, while failures attributed to nondiscrimination are 45% versus 17%. Task composition, provider coverage, attempts per task, rollout engine, and timeout budget all differ, and the audit judge has only 67% self-consistency under repeated judgment. More importantly, a post-hoc scorer audit found that many "exact" criteria checked strings in attachments copied into the scoring directory rather than in the produced artifact; those components could pass even when no deliverable was produced. The reported variance therefore primarily reflects structural checks and is diluted by constant attachment-based scores. It is not a measure of semantic deliverable quality.

On the 27 tasks with both rubric versions, a distinct cross-provider score-spread statistic changes from 0.341 to 0.326 rather than increasing. A paired rerun on identical tasks, policies, seeds, and a repaired scorer is required to estimate the causal effect of redesign.

Within the redesigned cohort, the ten highest-variance Qwen tasks have score ranges from 0.46 to 0.91 across five or six attempts. These cases are candidates for rerunning after scorer repair because a fixed policy exhibits structurally different outcomes, but variance alone does not establish rubric validity, factual quality, or downstream learnability.

5.3 Artifact mixing recovers score in a 30-step online pilot

Table 4 reports the 30-step GDPval-style pilot. Skill-only training slightly improves the skill holdout but reduces professional-artifact score from 0.160 to 0.056 and increases the number of zero-score tasks from 22 to 33. Mixing artifact tasks restores the skill score to roughly the base level and raises GDPval-style score to 0.237. On tasks receiving nonzero reward, quality rises from 0.329 to 0.486.

Table 4: Held-out artifact and primitive-skill results after a same-step 30-step online pilot. GDP-NZ is the mean among nonzero GDPval-style tasks.

Training	Skill	GDP mean	GDP-NZ	GDP zeros
Base	0.163	0.160	0.329	22 / 43
Skill only	**0.170**	0.056	0.242	33 / 43
Skill + artifact	0.161	**0.237**	**0.486**	**22 / 43**

This comparison does not isolate off-policy SFT—the pilot uses online updates—and it has one seed. It nevertheless identifies a distributional gap that WORKSYNTH targets: in this run, generic executable-skill training does not transfer to professional work, while mixing artifact tasks recovers and exceeds the base score.

5.4 ONLINE RL IMPROVES THE SCORER BUT REQUIRES REPEATED LONG ROLLOUTS

A 300-step GRPO run provides an online reference. On a fixed 18-task denominator with missing outputs scored zero, SkillBench increases from 0.163 to 0.289; on the 15 paired non-missing tasks it increases from 0.167 to 0.347. GDPval-style mean on all 43 tasks increases from 0.148 to 0.276. The run uses eight long sandbox rollouts per prompt and is sensitive to horizon choices. In an exploratory, noisy early-run control, raising the maximum tool turns from 12 to 20 is associated with a zero-reward rate of 59% rather than 66%, mean reward of 0.134 rather than 0.079, and fitted reward slope of 1.55×10^{-3} rather than 6.3×10^{-4} per step. Successful rollouts have a median of 18 turns, making horizon truncation a plausible explanation; the control requires a full-length replication.

These observations motivate trajectory reuse. Once produced, a trace can be replayed without coupling each gradient step to sandbox availability, judge latency, and stochastic group composition. We do not report an end-to-end compute or monetary-cost comparison between data creation plus SFT and online RL.

6 ANALYSIS

6.1 WHAT THE CONTROLLED COMPARISON IDENTIFIES

The cleanest positive result is the base-to-Random comparison: two epochs of masked behavior cloning on 6,500 frozen software-agent traces raise resolution from 1.6% to 9.8%. Full reaches 10.8%. These arms establish that logged trajectories can teach environment-facing behavior without an environment in the update loop.

Rule-selected (v1) is higher than Random at matched example count, and its paired instance-level difference is statistically distinguishable in the 9B run. Yet selection jointly changes source success, task and repository mix, trajectory length, tool behavior, and total tokens. The rule itself fails post-hoc validity checks, and the ordering reverses in the unmatched 27B check. The defensible conclusion is therefore specific: this frozen membership rule changes 9B outcomes, not that its five process metrics are valid rewards or that less data is generally better.

6.2 HYPOTHESES AND COMPETING EXPLANATIONS

Our working hypothesis is that useful work traces instantiate a reusable cycle: inspect state, produce an artifact, test it against requirements, repair a failure, and verify the result. Criterion-level outcomes could help select such traces, while critique observations already inside the interaction could make the repair cycle learnable. The present experiments do not identify either mechanism. The v1 selector's success gate is a simpler explanation for its 9B gain, and the professional rubric study is confounded by cohort composition and a flawed scorer.

These alternatives yield direct tests. To isolate revision-oriented supervision, one should hold tasks, final artifacts, and token budgets fixed while removing intermediate failures and repairs. To isolate criterion alignment, one should compare a validated selector with passed-only and random controls matched on task, repository, length, and teacher policy. To establish professional transfer, the final-

output, critique-and-revision, and genre-targeted variants must train the same checkpoint on the repaired professional corpus and evaluate on a fixed GDPval-style holdout.

6.3 Failure modes are intervention-specific

Observed professional-work failures include (i) horizon failure, where no deliverable appears before termination; (ii) execution failure, where a file is missing, malformed, or cannot be staged; (iii) content failure, where a valid artifact omits required facts or conventions; and (iv) measurement failure, where the scorer rewards copied attachments or incidental reference properties. These classes call for different interventions: adaptive horizons for (i), more reliable tools for (ii), work-oriented supervision for (iii), and evaluator repair for (iv). Collapsing them to one scalar obscures the intervention needed.

Off-policy training adds two risks. A trace may rely on teacher-specific actions or environment states the student cannot reproduce, and aggressive filtering may remove legitimate stylistic or structural diversity. A selector can amplify evaluator mistakes as readily as useful patterns; criteria must therefore be validated before they become data-membership rules.

7 Limitations and Responsible Interpretation

Our controlled off-policy evidence is limited to a 9B model trained on software-engineering trajectories. Software patches are executable professional artifacts, but they do not capture the visual, editorial, and organizational conventions of reports, spreadsheets, and slides. The professional branch demonstrates construction feasibility and exposes evaluator failures; the 30-step GDPval-style comparison uses online updates. It is not yet evidence that off-policy WorkSynth training improves GDPval itself.

The present experiments do not isolate the three supervision components named in the abstract. The professional synthetic corpus has not yet been used in a same-checkpoint offline final-output versus critique-and-revision versus genre-specific training matrix. Those comparisons remain hypotheses, not completed causal findings.

Each completed 9B SFT arm uses one training seed. Evaluation on all 500 SWE-bench Verified instances reduces test-set sampling uncertainty but not optimization variance. Rule-selected and Random are equal in trajectory count but differ by 6.0% in tokens and are not matched on task or repository. Repository-level overlap affects 53.8% of evaluation instances, although the Django-only result removes that confound for one large subset. The unmatched 27B reversal further limits generalization across model scales and training schedules.

The professional data has implementation limitations beyond judge bias. Only 48 of 83 verification reports pass the recorded integrity threshold, yet rollout launch was not strictly gated. Semantic-judge criteria were skipped in the large rollout run, and many exact checks inadvertently inspected copied attachments. Old and redesigned rubric summaries use different task pools and the trajectory-aware audit has limited self-consistency. A repaired evaluator, paired rerun, and independent expert review are prerequisites for claims about semantic deliverable quality.

Finally, reuse reduces repeated interaction during optimization but does not make data creation free. WorkSynth still requires teacher rollouts, sandbox execution, and evaluation before SFT, and we have not established a total-compute crossover point against online RL. Source documents and templates also require licensing, privacy, and provenance review before deployment.

8 Conclusion

Frozen agent trajectories substantially improve a 9B software agent, while a historical v1 subset performs best only in that setting and reverses against Full in an unmatched 27B check. The professional branch demonstrates construction feasibility, exposes scorer failures, and offers an online artifact-mixing signal—not completed off-policy GDPval evidence. WorkSynth therefore remains a testable program: validate artifacts and criteria, then compare final-output, critique-and-revision, and genre-targeted supervision under matched tasks, tokens, checkpoints, and a repaired evaluator.

AI Use Statement

Generative AI tools were used in two roles. First, as part of the methodology, language models generated synthetic professional tasks, reference artifacts, rubrics, and trajectories, and served as rollout policies and semantic evaluators; these uses are described in Sections 3–4. Second, generative AI assisted with literature discovery and synthesis, research framing, organization and drafting of the manuscript, preparation of LaTeX tables and figures, and inspection and summarization of saved experiment logs and JSON artifacts. AI-suggested citations and quantitative claims were checked against public sources and saved experimental records during preparation. Generative AI was not treated as an author or authoritative source. The authors retain responsibility for final verification and for all scientific and editorial decisions.

Ethics Statement

The reported experiments use synthetic professional tasks, model-generated trajectories, public benchmarks, and third-party language models; they do not recruit human participants or make decisions about individuals. Professional source documents and generated artifacts may nevertheless contain private, proprietary, copyrighted, inaccurate, or biased content. Only redistributable material should be released, with privacy and license review for source data and hashed manifests or interface specifications for restricted resources. Rubric-based and LLM-based evaluators can reproduce domain and model biases, and their scores are not substitutes for expert review. The present results are not evidence that artifact agents are ready for unattended deployment in high-stakes professional settings.

Reproducibility Statement

Sections 3–4 specify the task representation, construction stages, training objective, datasets, models, and evaluation protocols. The appendix records dataset counts, thresholds, optimizer settings, split construction, contamination tests, and claim boundaries. Reported values were cross-checked against the source summaries, result files, and scripts listed in the internal claim-to-evidence map. The anonymous supplementary material should include task manifests, rubric schemas, evaluation scripts, configurations, and aggregate results; model endpoints and source documents that cannot be redistributed can be represented by interface specifications and hashed manifests.

References

Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. Agent-FLAN: Designing data and methods of effective agent tuning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 9354–9366. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.557. URL https://aclanthology.org/2024.findings-acl.557/.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.06770.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, JD Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/871ac99fdc5282d0301934d23945ebaa-Abstract-Conference.html.

Zeyao Ma, Bohan Zhang, Jing Zhang, Jifan Yu, Xiaokang Zhang, Xiaohan Zhang, Sijia Luo, Xi Wang, and Jie Tang. SpreadsheetBench: Towards challenging real world spreadsheet manipulation. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3007. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ac840df270ac537dd74530a15c332684-Abstract-Datasets_and_Benchmarks_Track.html.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2303.17651.

Samuel Miserendino, Michele Wang, Tejal Patwardhan, and Johannes Heidecke. SWE-lancer: Can frontier LLMs earn $1 million from real-world freelance software engineering? In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 44412–44450. PMLR, 2025. URL https://proceedings.mlr.press/v267/miserendino25a.html.

Arindam Mitra, Luciano Del Corro, Guoqing Zheng, Shweti Mahajan, Dany Rouhana, Andres Codas, Yadong Lu, Wei-ge Chen, Olga Vrousgos, Corby Rosset, Fillipe Silva, Hamed Khanpour, Yash Lara, and Ahmed Awadallah. AgentInstruct: Toward generative teaching with agentic flows, 2024.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2203.02155.

Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. Training software engineering agents and verifiers with SWE-Gym. In *International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2412.21139.

Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simon Fishman, Marwan Aljubeh, Phoebe Thacker, Laurance Fauconnet, Natalie Kim, Samuel Miserendino, Gildas Chabot, David Li, Patrick Chao, Michael Sharman, Alexandra Barr, Amelia Glaese, and Jerry Tworek. GDPval: Evaluating AI model performance on real-world economically valuable tasks. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/290c2430f91912204f30bbcc990fff1d-Abstract-Conference.html.

Qwen Team. Qwen3.5-9B. Hugging Face model card, 2026. URL https://huggingface.co/Qwen/Qwen3.5-9B.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.18290.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. PaperBench: Evaluating AI's ability to replicate AI research. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 56843–56873. PMLR, 2025. URL https://proceedings.mlr.press/v267/starace25a.html.

Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Daniel Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, Junyang Lin, Robert Brennan, Hao Peng, Heng Ji, and Graham Neubig. OpenHands: An open platform for AI software developers as generalist agents. In *International Conference on Learning Representations*,

2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/a4b6ad6b48850c0c331d1259fc66a69c-Abstract-Conference.html.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Annual Meeting of the Association for Computational Linguistics*, 2023. URL https://arxiv.org/abs/2212.10560.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2404.07972.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.03629.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. AgentTuning: Enabling generalized agent abilities for LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3053–3077. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.181. URL https://aclanthology.org/2024.findings-acl.181/.

Hao Zheng, Xinyan Guan, Hao Kong, Wenkai Zhang, Jia Zheng, Weixiang Zhou, Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. PPTAgent: Generating and evaluating presentations beyond text-to-slides. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 14402–14418. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.728. URL https://aclanthology.org/2025.emnlp-main.728/.

A Dataset and Training Details

A.1 Professional-work task schema

Each completed task directory may contain a brief, attachments, a reference deliverable, a rubric, and a verification report; the inventory is incomplete at intermediate stages as quantified in Table 1. Rollouts are stored separately by provider and contain messages, tool calls, timing, termination reason, produced files, scalar score, and criterion outcomes. This separation prevents an agent from reading the reference artifact during evaluation.

The original criteria distribution targeted approximately 30% structural checks, 45% exact or reference-grounded facts, 20% semantic quality, and 5% penalties. The redesigned generator adds negative examples for brittle criteria, forbids hard-coded output filenames when the prompt does not require one, and normalizes Unicode whitespace and dash variants before exact comparison.

A.2 Professional-work rollout statistics

On the 12 tasks with at least three valid rollouts from all four policies, mean scores are 0.297 (Qwen), 0.220 (GLM), 0.150 (Kimi), and 0.110 (DeepSeek). File-production rates are 38%, 26%, 15%, and 10%, respectively. Provider infrastructure is not matched: slower endpoints experience more stream timeouts, so these numbers should not be interpreted as a clean model ranking. The within-policy variance comparisons in Table 3 are less sensitive to this issue.

A.3 Software trajectory filtering

The raw index contains 95,828 trajectories, of which 55,131 are marked successful by the source environment. The historical v1 membership rule is

$$\begin{aligned} &\text{pass} \ \wedge\ \text{reasoning_coverage} \geq 0.64 \ \wedge\ \text{post_fix_verify} \geq 1 \ \wedge \\ &\text{blind_retry_ratio} = 0 \ \wedge\ \text{format_fail_rate} \leq 0.05 \ \wedge\ \text{effective_step_rate} \geq 1.0. \end{aligned} \tag{7}$$

After holding out evaluation candidates, tokenizing with the student chat template, imposing the 65,536-token limit, and validating tool-call arguments, 6,500 Rule-selected (v1) trajectories remain. Random is sampled by a frozen NumPy seed 0 permutation from the same trainable universe. Full contains all 16,728 trainable trajectories. The post-hoc audit summarized in Section 5 limits interpretation of the rule.

Table 5: Token statistics for off-policy SFT arms. Percentiles count rendered student-model tokens.

Arm	n	p10	p50	p90	p99	Total
Rule-selected (v1)	6,500	36.9k	54.1k	63.7k	65.3k	338.0M
Random	6,500	32.3k	51.0k	62.9k	65.3k	318.8M
Full	16,728	31.8k	51.0k	62.9k	65.3k	819.4M

A.4 SFT hyperparameters

All 9B SFT runs use tensor parallelism 4, sequence parallelism, no context or pipeline parallelism, full activation recomputation, dynamic packing up to 66,560 tokens per GPU, and assistant-only loss. The optimizer is Adam with $\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0.1, peak learning rate 10^{-5}, cosine decay to 10^{-6}, and 10% warmup. Global batch size is 128. Dropout is disabled. Each arm is trained for two epochs.

B Evaluation Details

B.1 SWE-bench Verified

The agent is allowed at most 40 interaction turns. It edits a repository in an isolated sandbox and returns a patch. The standard task tests determine resolution. We checkpoint results per instance and

retry infrastructure errors without overwriting completed examples. Table 2 counts each of the 500 benchmark instances exactly once.

B.2 Contamination audit

We compare 45,170 unique training instance IDs against 500 SWE-bench Verified IDs and observe zero exact overlap. We then normalize issue descriptions by lowercasing, removing URLs, folding punctuation and whitespace, and compute character 5-gram Jaccard candidates with 128-permutation MinHash. No training–evaluation pair has exact similarity at or above 0.8. The sole pair between 0.6 and 0.8 is a false positive caused by a shared scikit-learn issue template and describes different issues.

At repository granularity, 11 of 12 evaluation repositories appear in training. The exception is Django, which accounts for 231 evaluation instances. On this zero-repository-overlap subset, Base, Random, Full, and Rule-selected (v1) resolve 3, 12, 20, and 30 tasks, respectively; Table 2 reports the corresponding rates.

B.3 GDPval-style scoring

The split contains 220 tasks, with 43 held out by an occupationally stratified seed-42 partition. The scorer first checks whether the artifact can be located and parsed, then runs deterministic criteria, then sends extractable content and semantic criteria to a fixed judge. We distinguish a model-produced zero from an evaluator failure; the latter is recorded as missing and retried. The 30-step table uses the same split and scoring code for all arms.

C Additional Online-RL Diagnostics

The online run illustrates why compact-answer RL recipes do not transfer mechanically to artifacts. Between 60% and 80% of early rollouts receive zero reward. Inspection attributes these zeros to missing deliverables, unsupported media extraction, broken staging, and tasks outside the policy's competence band. An all-zero eight-sample prompt group supplies no relative advantage. A preliminary horizon comparison is consistent with 20 turns helping relative to 12 because successful work takes a median of 18 turns, but the noisy early-run result requires replication.

Long runs additionally depend on sandbox availability, document libraries, parser compatibility, service timeouts, and judge endpoints. None of these issues invalidates online RL, which eventually produces strong improvements, but they tightly couple data ablations to external execution. Off-policy training makes task composition and dataset membership independently testable.

D CLAIM-TO-EVIDENCE MATRIX

Table 6: Scope of the empirical claims. This table is included to distinguish completed causal comparisons from supporting analyses.

Claim	Evidence in this paper	Boundary
Frozen-trajectory SFT improves the 9B software agent	Base 1.6%; Random 9.8%; Full 10.8% on all 500 tasks	One seed and one software-artifact checkpoint
Rule-selected (v1) is highest in the 9B run	14.4% vs. Random 9.8%, paired CI $[1.6, 7.6]$, $p = .004$	Flawed historical rule; not token/task matched; unmatched 27B ordering reverses
The zero-repository-overlap subset preserves the 9B ordering	Django-only: 1.3%, 5.2%, 8.7%, 13.0% for Base/Random/Full/Rule	One repository and one seed; Rule vs. Full $p = .087$
Pipeline/rubric revision changes observed diagnostics	Separate old/new cohorts summarized in Table 3	Multiple confounds; semantic judge skipped; exact-score leakage; not downstream training evidence
Artifact mixing recovers the online pilot score	Same-step 43-task comparison: 0.160 base, 0.056 skill, 0.237 mixed	One-seed online pilot; not professional-corpus off-policy SFT
Explicit critique tokens are causally beneficial	Not isolated	Requires matched traces with identical tasks and finals, removing only critique/repair context