TEST-TIME SELF IMPROVEMENT BY SCALING UNSUPERVISED WORLD INTERNALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Agents accumulate experience as they interact with an environment, but current systems typically reuse this experience through context or external memory while keeping the underlying model fixed. While flexible, this constrains the form of learning models can do especially under novel and complex environments. We ask whether weight updates can instead improve the model's ability to solve tasks by learning directly from its own experience, without feedback on whther its attempts succeed. We study *world internalization*, a training approach that converts information acquired through unsupervised exploration into parameter updates. World internalization trains on insights and summaries extracted from exploratory trajectories, together with a small fraction of trajectories that teach the model to integrate newly acquired knowledge with existing cognitive priors for reasoning and action. This enables the model to internalize environmental knowledge through continued training analogous to pretraining and midtraining while preserving its ability to use that knowledge for downstream tasks. We evaluate world internalization across three domains with distinct forms of environmental knowledge and interaction: Harvey's Legal Agent Benchmark, StudyBench, and Equational Theories. We observe that internalization reliably improves downstream task success despite receiving no supervision on improving task solving. Moreover, performance scales nearly log-linearly as internalization data increases from 10M to 100M tokens. In all, our results show that scaling the internalization of autonomous experience can enable self improvement.

1 INTRODUCTION

Large language models (LLMs) have become increasingly capable of solving long-horizon tasks by spending more test-time compute. To do so, modern systems augment a model with increasingly sophisticated harnesses that come equipped with memory, compaction, and subagents. These mechanisms allow an agent to make better use of experience without changing the model itself. However, this improvement is fundamentally mediated by the test-time harness: the underlying post-trained model remains fixed, and accumulated experience must continually be compressed, retrieved, and placed back into context. Moreover, gains from simply scaling test-time compute eventually exhibit diminishing returns for current models (Zhu et al., 2026), while models remain imperfect at using increasingly large contexts (Liu et al., 2024). This raises a broader question: *can a deployed model self-improve as it accumulates experience in a new environment?*

In this work, we study whether experience acquired during deployment can be converted into persistent updates to the model's weights as knowledge it can draw on to reason and act in downstream tasks. Specifically, we consider a problem setting that is "unsupervised" with respect to downstream task solving: the agent is not given the downstream questions on which it will later be evaluated, demonstrations of how to solve them or any reward functions. Instead, it autonomously explores an environment and learns from the information it encounters. We call this process ***world internalization***. The goal is therefore different from compressing context for a particular input or optimizing a policy against a known objective. We ask whether internalizing an environment into the weights can make a model better at generally solving tasks within it.

Doing so is nontrivial because experience collected through interaction is not naturally a useful training distribution. A raw trajectory would contain failed tool calls, terminal outputs, copied code, redundant observations, and other artifacts of the model's current policy. Running direct next-

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.

token prediction on such trajectories can reinforce poor behavior, result in forgetting, and amplify memorization of spurious patterns rather than capture what was learned. Even training directly on the raw information observed by the agent need not make that information reliably actionable after training (Allen-Zhu & Li, 2024; Ovadia et al., 2024). Our first insight is therefore to exploit a capability that instruction-tuned models already possess: distilling long and noisy contexts into compact, semantically meaningful representations. After exploration, the agent produces grounded notes that capture *what it learned* from its experience. These notes turn noisy interactions into a scalable target for continual weight updates, allowing the model to internalize knowledge it sees.

Yet acquiring knowledge is only a part of the overall problem. A model may store new information without learning when or how to use it during reasoning and action (Allen-Zhu & Li, 2024). We therefore construct a second, midtraining-like data source: the frozen model generates synthetic questions for itself and solves them while conditioned on the summaries produced during exploration. We instruct it to use this information naturally, without referring to the notes explicitly, and remove the note tokens during training. This yields a form of context distillation in which knowledge acquired by spending inference-time compute is distilled into the model's behavior. The two data sources thus play complementary roles: distilling summary documents teaches the model "what is true about the environment", while distilling knowledge-conditioned trajectories teaches it "how to combine learned knowledge with reasoning and action". Crucially, these trajectories can be generated entirely by the base model, exploiting an *asymmetry* between its ability to follow instructions to propose diverse synthetic tasks and exploratory behaviors, and its ability to precisely solve those tasks (Setlur et al., 2026). We refer to this training procedure as U-WIN.

Perhaps surprisingly, this recipe yields self-improvement without directly supervising improved task solving. Our training procedure never uses downstream evaluation questions, receives rewards for solving them, or filters for successful trajectories. Instead, the model improves by internalizing environmental knowledge and learning how to retrieve and compose it with capabilities already present in the post-trained base model. This distinguishes world internalization from RL approaches that optimize explicit task-solving trajectories (Zelikman et al., 2022; Singh et al., 2024; Huang et al., 2026), and from synthetic continued pretraining, which transforms an externally supplied corpus into a richer training distribution (Yang et al., 2025; Maini et al., 2024).

We evaluate U-WIN on C&H (Harvey AI, 2026), StudyBench (Li et al., 2026), and Equational Theories (Bolan et al., 2025), using Qwen3.5-9B and Qwen3.5-27B base models. Across all environments, downstream performance improves despite no access to evaluation tasks or correctness signals during data generation, and continues to generally improve with scale. Type I notes are effective targets for internalizing environmental knowledge, while Type II trajectories teach the model to surface and use that knowledge during reasoning and action. Together, these results suggest a path toward continual learning from deployment experience, where autonomous interaction can generate not only context for a fixed model, but scalable training targets that allow the model itself to progressively internalize its environment and improve on previously unseen downstream tasks.

2 Related Work

Learning from test-time experience without weight updates. Most agents accumulate experience outside the model, as reflections, workflows, skills, or explicit memory (Shinn et al., 2023; Wang et al., 2025; 2023; Packer et al., 2023). Other work optimizes the harness against a task distribution, evolving prompts and contexts from task feedback (Agrawal et al., 2026; Zhang et al., 2026; Suzgun et al., 2025), searching over context-management code (Lee et al., 2026), or learning to consolidate experience into compact states (Zhou et al., 2026; Wu et al., 2026). These mechanisms extend the effective horizon of a fixed model, but accumulated knowledge must still be selected, compressed, and retrieved into context. Long-context models do not reliably use all information in context (Liu et al., 2024), and gains from prolonged environment interaction show diminishing returns for current agents (Zhu et al., 2026). We instead use test-time experience to update the model itself, so that knowledge acquired across interactions accumulates in its parameters.

Continued pretraining and test-time training. At the optimization level, U-WIN is related to continued pretraining (CPT) on new domains (Gururangan et al., 2020). Because training on raw information does not ensure that knowledge is extractable (Allen-Zhu & Li, 2024; Ovadia et al., 2024), synthetic CPT rewrites a supplied corpus into a larger, more diverse training set (Maini et al., 2024; Yang et al., 2025; DatologyAI et al., 2025). In U-WIN, no corpus is supplied: the documents

are the agent's own notes on exploration it chose, so errors in what it explores and concludes enter the training data. Unlike recursive training on model outputs (Shumailov et al., 2024), the notes are grounded in environment observations and generated by the frozen base model, so training adds environmental information rather than resampling the model's own distribution. Test-time training (TTT) also updates weights at inference time, but adapts to a single long context (Tandon et al., 2025), uses downstream performance as reward (Zweiger et al., 2025), or optimizes a single test problem with an explicit reward (Yuksekgonul et al., 2026). We instead internalize experience from interactions, without reward and *before* the downstream tasks are known.

Self-improvement from self-generated data. Many methods train on self-generated solutions filtered or reinforced by correctness feedback (Zelikman et al., 2022; Singh et al., 2024; Liu et al., 2025b; Setlur et al., 2026), or extract denser supervision from interaction through privileged self-teachers (Zhao et al., 2026), textual environment feedback (Hübotter et al., 2026), or observation prediction (Shrivastava et al., 2026). Eyuboglu et al. (2026) train KV cache "cartridges" on self-generated data but their goal is distill long contexts, while our goal is to accumulate expertise in a the environment. ECHO (Shrivastava et al., 2026) is closest to our setting but trains on raw observation tokens from rollouts on a given task distribution, whereas U-WIN trains on notes and knowledge-conditioned trajectories for self-proposed tasks. Other methods jointly adapt weights and harness against a rewarded task family (Tiwari et al., 2026; Kim et al., 2026; Luo et al., 2026) or replace human-provided questions with self-play (Huang et al., 2026; Liu et al., 2025a), but still organize learning around explicit tasks and success signals. World internalization uses no oracle questions, solutions, or rewards; the model generates supervision for *what it learns about the environment*, and we ask whether better task solving emerges from scaling this internalized knowledge.

3 Problem Setup and Definition

Environment and interaction protocol. We consider an LLM-based agent π_{base} interacting with an environment $\mathcal{E}$. The environment consists of a knowledge source or corpus together with a set of actions through which the agent can explore it. For example, $\mathcal{E}$ may be a collection of documents that can be searched and read, an unfamiliar codebase that can be inspected, or a structured mathematical domain that can be queried. Interaction produces trajectories $\tau = (o_0, a_0, \ldots, o_T)$, where o_t denotes information observed from the environment and a_t denotes the agent's reasoning, tool call, or action. Importantly, $\mathcal{E}$ contains knowledge that is not readily available to π_{base} beforehand and must therefore be acquired through interaction.

World internalization. We study how an agent can learn useful expertise about an environment, a process we call *world internalization*. An LLM agent autonomously collects interaction trajectories $\mathcal{T}_{\mathcal{E}} = \{\tau_i\}_{i=1}^{N}$ from $\mathcal{E}$ and uses this experience to update π_{base} into an internalized model π_{int}. Crucially, the agent has *no* knowledge of the downstream tasks on which π_{int} will be evaluated, nor access to rewards or other signals of task success. It must therefore decide what information to acquire and internalize broadly useful knowledge rather than optimize toward a known task distribution. ***Our central question*** is whether scaling autonomously acquired experience with today's capable post-trained models can continually improve performance on previously unseen tasks in the same environment. Doing so requires priors for identifying useful information, exploring diverse behaviors, and incorporating new knowledge without forgetting existing capabilities.

Relationship with continual and test-time learning. Our setting can be viewed as a form of continual or test-time learning, but differs primarily in what experience is assumed to be available for learning. Standard continual learning typically considers a sequence of tasks or data distributions presented to the learner (Van de Ven et al., 2022). Recent test-time training methods similarly specify a concrete adaptation signal, such as a given long context to compress (Tandon et al., 2025) or an explicit reward for a particular problem (Yuksekgonul et al., 2026). In contrast, we assume only access to an environment $\mathcal{E}$ and a set of tools. The agent must itself decide what to explore, what experience to collect, and what information to internalize before the downstream tasks are known. Thus, in our setting, generating useful experience is itself part of the learning problem.

4 U-WIN: Unsupervised World Internalization

In this section, we introduce U-WIN, our approach for converting experience gathered through *unsupervised* interaction into persistent knowledge and improved downstream behavior. Doing so requires addressing three challenges: **(1)** collecting informative experience from an environment, **(2)**

transforming that experience into a form that can be effectively internalized by the model, and **(3)** teaching the model to use the acquired knowledge during reasoning and action. We construct two complementary forms of internalization data. **Type I data** consists of synthetic notes distilled from the agent's exploration experience and is intended to encode knowledge about the environment. **Type II data** consists of knowledge-conditioned trajectories that demonstrate how this acquired knowledge can be recalled and combined with reasoning and action. We train on these two data sources using next-token prediction, allowing the model to both acquire environmental knowledge and connect that knowledge to capabilities already present in the base model. We describe the construction and training of each data type below.

4.1 Type I Data: Internalizing Knowledge from Synthetic Exploration

We begin by collecting agentic trajectories through unsupervised interaction with the environment. Doing so requires giving the model some objective that induces it to act. Rather than supplying downstream tasks, we ask the model itself to synthesize tasks that it can attempt. These tasks need not always be correct or even solvable. Their purpose is simply to induce a diverse set of behaviors and interactions that expose the model to different parts of the environment.

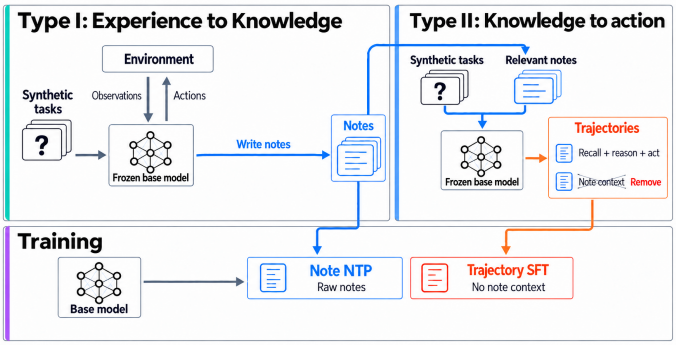


Figure 1: ***Overview of* U-WIN.** We generate two complementary types of synthetic data using a frozen base model in a new environment, without access to downstream test questions. **Type I: Experience to Knowledge**. The agent explores the environment on synthetic tasks and writes notes capturing what it learns. **Type II: Knowledge to Action.** We provide relevant notes in context and generate trajectories that interleave recalling this knowledge with reasoning and action. We then train on the note documents through next-token prediction and on the trajectories with the note context removed, after mixing with data of Type I.

Generating base note documents. Given a set of seed tasks, we organize this process into multiple rounds. In each round, starting from a set of synthetically proposed tasks, the model acts in the environment and collects trajectories. Once this experience has been gathered, we ask the same base model to inspect one or more trajectories, identify concrete "insights" it learned, surprising observations, or reusable information, and write a document about these in natural language. This form of a *note* document may summarize what behavior was attempted, compare information across trajectories, record where a functionality appears in the environment, or identify which tools or actions were useful or redundant. We intentionally keep the definition of a note high-level so that the resulting documents remain diverse in both content and structure. Since the base model is already instruction tuned, we find that it can reliably transform its interaction experience into such note documents.

Recursive scaling of note tokens. We scale this process over many rounds. Later rounds are *recursive*: in addition to interacting with the environment, the agent is given access to notes written in previous rounds and can search over them during its unsupervised exploration. This allows subsequent interaction to build on previously acquired knowledge, investigate unresolved questions, make connections across experiences, or revise earlier conclusions. Repeating this process produces a growing corpus of synthetic note documents. These notes provide a compressed representation of the agent's interaction experience. In practice, we observe that this compression can be substantial: for example, trajectories spanning many interaction turns and tens of thousands of reasoning tokens can often be distilled into note documents containing only hundreds to roughly a thousand tokens.

Training on note documents. Once a sufficient number of note tokens has been collected, we can train the model directly on this corpus using next-token prediction. Since the starting model is already post-trained, one might expect that preserving a chat or reasoning template would be important for effective subsequent training. We find, however, that this is not necessary. Instead, we treat the notes as ordinary "pretraining" documents and minimize a standard negative log-likelihood objective over their tokens, with explicit thinking disabled. Naïvely optimizing this objective can, however, lead to catastrophic forgetting of the model's existing reasoning and action capabilities. To

mitigate this, we additionally regularize the updated model toward the base model using a forward KL-divergence loss on a small held-out set of agentic trajectories generated by the base model. We will discuss our concrete loss objective after discussing the second type of data that we also use.

4.2 Type II Data: Connecting Internalized Knowledge to Action

Distilling note documents teaches the model knowledge about the environment, but does not necessarily teach it how to use this knowledge while solving a task. Although some notes summarize useful behaviors or strategies, the fraction of "actionable" information from Stage I is generally much smaller than the factual content. Indeed, past work shows that next-token prediction (NTP) on factual information alone need not make that information reliably accessible for downstream behavior (Allen-Zhu & Li, 2024; Ovadia et al., 2024). We therefore construct a second source of synthetic data that illustrates how knowledge acquired in Stage I can be recalled and "stitched" into the model's existing reasoning, action, and tool-use capabilities. We view this as analogous to *mid-training* in LLM pipelines: rather than primarily introducing new knowledge, it instills behaviors for integrating newly acquired knowledge with capabilities already present in the post-trained model.

To generate this midtraining-like data, we prompt the frozen base model to solve a uniformly sampled subset of the synthetic questions from note generation, with a mixture of relevant and irrelevant notes in context. The notes are *privileged*: the model should use them whenever they help direct its search and may present their contents as recalled knowledge, but must never mention the notes (see Fig 7). The resulting trajectories interleave reasoning and actions with facts and strategies acquired during exploration. We then train on these trajectories *without* the notes in context.

This resembles context distillation (Snell et al., 2022): the teacher generates behavior with access to privileged context, while the student is trained to reproduce that behavior without it. Here, however, the privileged context consists specifically of knowledge that the agent itself is expected to acquire from Type I data. Distilling these trajectories therefore encourages the model to connect representations that encode this learned knowledge with the reasoning and action patterns required to solve related tasks. In this sense, the two types of data play complementary roles: Type I teaches the model *what it learned about the environment*, while Type II teaches it *how to use what it learned*.

Crucially, this stage still does not use downstream task supervision. We neither use the downstream evaluation questions to generate these trajectories, nor filter trajectories according to whether their synthetic tasks were solved successfully. In principle, poor synthetic trajectories could degrade the model's behavior. In practice, we use relatively simple synthetic questions for which the base model can generally produce reasonable behavior. These examples need only teach the basic operation of recalling environmental knowledge and combining it with reasoning and action. Once learned, this operation can transfer to substantially harder downstream tasks that require longer reasoning horizons and many more interactions with the environment.

Implementation details: **training on note documents and Type II data.** In practice, we can train the two data sources sequentially, mimicking pretraining followed by midtraining, or mix note documents and knowledge-conditioned trajectories within a single run; we use the latter by default. To limit drift from the base model, we additionally **(1)** sample trajectories from the base model on synthetic questions and maximize their likelihood during training, and **(2)** linearly merge the final model with the base model (Wortsman et al., 2022). We find that model merging not only preserves acquired knowledge and the search behaviors learned from Type II data, but also helps integrate this knowledge with broader behaviors already present in the base model, such as search-verify-revise, which may be underrepresented in the Type II data.

5 Experimental Setup and Details

To evaluate U-WIN, we pick three environments that contain substantial knowledge and strategies not already available to the base model. In particular, we seek environments that test whether newly acquired knowledge can be recalled directly, combined with tool use and action, and leveraged to support solving downstream tasks over the environment. To this end, we consider the Harvey Legal Agent Benchmark (Harvey AI, 2026), StudyBench (Li et al., 2026), and Equational Theories (Bolan et al., 2025). Together, these domains require models to acquire, integrate, and apply knowledge across tasks ranging from question-answering to agentic interaction and exploration.

Harvey Legal Agent Benchmark (or "C&H") (Harvey AI, 2026). This benchmark instantiates a synthetic law firm, C&H, as a persistent file system containing emails, legal documents, memos,

spreadsheets, and other internal records. During exploration, the agent can search and read files, as well as create, read, and edit notes containing useful information it discovers. For evaluation, we first use a closed-book knowledge-retention suite containing roughly 8,000 probes spanning factual questions, reverse questions, multiple-choice statements, and false-statement rejection. These probes are evaluated without access to the environment or exploration notes, allowing us to directly measure how much environment-specific knowledge has been internalized into the model's parameters, but not necessarily whether the agent can use this knowledge to take the right actions. We therefore additionally evaluate on the benchmark's 250 downstream agentic tasks, which primarily require finding relevant evidence and combining information across documents. During this evaluation, the model can search and read the original environment but does not have access to its exploration notes. These two evaluations allow us to distinguish between *acquiring* knowledge about the environment and *using* that knowledge during downstream agentic behavior.

Equational Theories. The Equational Theories Project (Bolan et al., 2025) studies implication relationships among 4,694 equational laws over magmas, giving rise to over 22 million possible ordered implications. The environment therefore provides a large body of mathematical knowledge in which individual relationships can be learned and composed. We test whether internalization generalizes to *unseen implication relationships*: given two equations, the model must determine whether the first implies the second. We use four released evaluation subsets[1], each containing 200 problems with balanced true and false labels. This setting tests whether the model can use internalized math structure to correctly reason about relations it did not observe during training.

StudyBench. StudyBench (Li et al., 2026) evaluates whether agents can acquire expertise in novel domains by purely studying a given corpus. We use the OpenClaw and DSPy environments, where the codebase files represent the documents to study. Evaluation consists of 20 source-grounded coding questions that require the model to configure, extend, or reason about the codebase. We evaluate these questions in two settings: a *direct* setting, where the model must answer without tool access, and an *agentic* setting, where it can interact with the codebase using tools for up to 20 turns. Together, these evaluations measure both how much codebase knowledge has been internalized and how effectively that knowledge can be used during downstream problem solving.

Training Details. For C&H and Equational Theories, we use Qwen3.5-9B as the base model and use the same frozen checkpoint to generate all internalization data. For StudyBench, we use Qwen3.8-27B, primarily for diversity in model size within our experiments and its agentic capabilities to navigate and understand content from codebases. We jointly train on Type I and Type II data in a single run, with approximately 70% of supervised tokens coming from Type I notes and 30% from Type II trajectories for C&H and Equational theories, and approximately 90% Type I tokens and 10% of Type II tokens for the StudyBench environments. We consider supervised-token budgets of 1M, 5M, 10M, 30M, and 100M faor C&H, and 1M, 3M, 10M, and 30M for Equational Theories. All models are fully finetuned for two epochs using a learning rate of 5×10^{-6} with cosine decay and 3% warmup. Full optimization and data-generation details are provided in Appendix A.1.

Inference Details. Within each environment, we use identical evaluation harnesses and inference settings for both the base and internalized models. For C&H, we evaluate closed-book performance deterministically using sequence log-probabilities over 1,780 multiple-choice probes. For Equational Theories, we sample four responses per problem using temperature 1.0 and nucleus sampling with $p = 0.95$, and report mean per-response success. During this evaluation, the model has access to neither the exploration notes nor the source environment. For StudyBench, we consider two evaluation settings. In the non-agentic setting, the model answers each question directly without taking actions; we provide a budget of 65,536 tokens and sample using the recommended Qwen3.8-27B settings with temperature 0.6. In the agentic setting, we provide a total budget of up to 10 tool calls within the 20 turns, with each turn allocated up to 16K reasoning tokens. Additional decoding parameters, prompts, and environment-specific evaluation details are provided in Appendix A.2.

6 Empirical Results and Ablation Experiments

The defining challenge of our setting is that the model does not know the downstream tasks on which it will eventually be evaluated. During exploration, it receives neither evaluation questions nor feedback indicating whether its actions on synthetic tasks are successful. We therefore ask three questions: **(1)** Does downstream performance improve as we scale the amount of world internaliza-

[1] https://huggingface.co/datasets/SAIRfoundation/equational-theories-selected-problems

tion? **(2)** Are Type I notes an effective target for knowledge internalization? and **(3)** Is Type II data necessary to preserve agentic behavior and connect internalized knowledge to action?

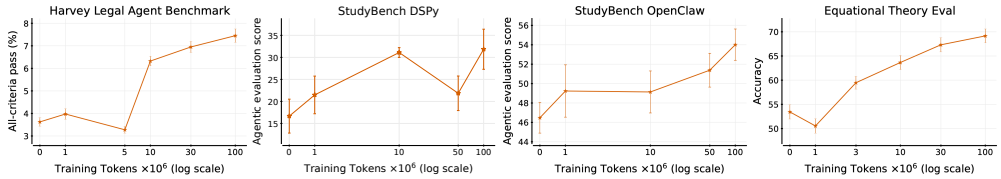


Figure 2: **Downstream task performance improves as world internalization scales.** Performance on C&H, StudyBench DSPy, StudyBench OpenClaw, and Equational Theorie as internalization data increases.

Result 1: Downstream task performance improves by scaling unsupervised world internalization. We first ask whether experience collected without any task-level correctness signal can nonetheless improve downstream performance. During exploration, the model never sees the downstream evaluation tasks, and even the synthetic tasks used to induce interaction are not scored, verified, or filtered for successful completion. Figure 2 shows that this is the case across all four environments, although the onset of improvement differs across domains. On C&H (left), performance remains close to the base model at small data scales before increasing sharply beyond 5M tokens and continuing to improve thereafter. Both the StudyBench domains (middle) show gains earlier, followed by further improvement as internalization data scales. Equational Theories (right) similarly exhibits a small low-data degradation followed by consistent gains at larger scales. Thus, while small amounts of self-generated experience are not always sufficient to improve downstream behavior, sufficiently large-scale internalization reliably surpasses the performance of the corresponding base model across qualitatively different forms of reasoning and action. Together, these results suggest that scaling world internalization enables knowledge acquired from autonomous interaction to generalize beyond the particular synthetic tasks used to collect that experience.

6.1 Type I data internalizes environmental knowledge

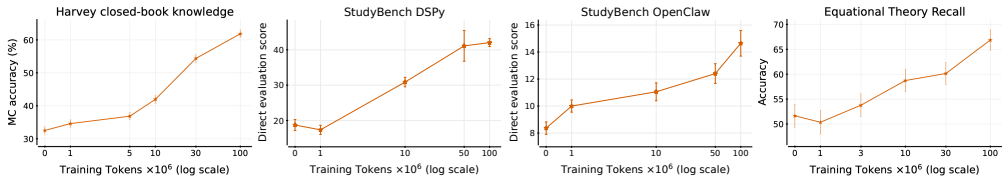


Figure 3: **Unsupervised world internalization increases parametric knowledge.** Closed-book performance vs. internalization data on C&H (left), StudyBench DSPy and OpenClaw (middle), and Equational Theories (right). Models answer without notes, tools, or environment access, so knowledge must be from the parameters.

U-WIN increases parametric knowledge. We next ask whether improvements in downstream performance are accompanied by increased knowledge of the environment stored directly in the model's parameters. We isolate this effect using *direct knowledge evaluations*, in which the model must answer questions without access to exploration notes, tools, or the underlying environment. Figure 3 shows that performance on these evaluations improves substantially as we scale the amount of internalization data. On C&H, closed-book knowledge increases sharply at larger training scales, while StudyBench shows steady improvement across the range considered. Because these evaluations target environment-specific information that cannot be acquired again at inference time, the observed scaling provides direct evidence that world internalization progressively transfers knowledge acquired through exploration into the model's parameters.

Type I notes provide an effective target for knowledge internalization. We use C&H for controlled ablations to understand which training targets most effectively internalize environmental knowledge. We first isolate Type I data and train solely with next-token prediction on the generated notes. As shown in Figure 4 left, notes-only training substantially improves closed-book knowledge with scale and, at the largest training budget, recovers most of the parametric knowledge gains obtained by joint Type I and Type II training. This indicates that Type I notes themselves are sufficient to drive substantial knowledge acquisition. We next ask whether this improvement arises simply from additional exposure to environment-specific information or depends on how that information is represented during training.

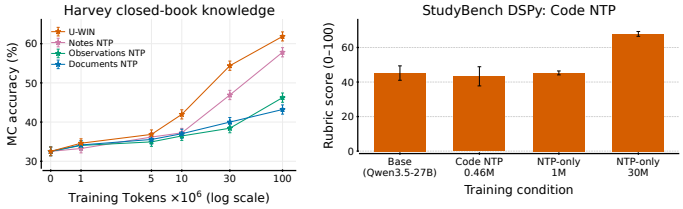


Figure 4: **Left.** Closed-book C&H knowledge accuracy as training data scales for U-WIN training, Type I notes, sampled environment documents, and interaction observations, all matched by token budget. **Right.** StudyBench agentic evaluation score vs training condition .

To answer this, we compare token-matched training on Type I notes against two alternative targets from the same environment: sampled excerpts from the underlying source documents and observations encountered during exploration. While all three targets provide access to environment-specific information, training on notes yields substantially stronger knowledge retention as scale increases, with the gap becoming especially pronounced at larger training budgets. In contrast, training directly on documents or observations produces considerably weaker gains. These results suggest that effective knowledge internalization depends not only on exposing the model to environmental information. StudyBench-DSPy provides a complementary setting where the underlying codebase contains only roughly 0.33M tokens. Direct continued training on this finite source provides little benefit, while Type I notes can be generated at substantially larger scale from interaction with the same environment. Figure 4 (right) shows that scaling this note corpus eventually produces a large improvement in downstream performance. Together, these results suggest that Type I notes serve two roles: they provide a more effective representation of environmental knowledge and turn a finite environment into a scalable source of training data. We also run an extended version of this experiment on the StudyBench DSPy environment. To isolate the effect of training on note documents, we compare against next-token prediction (NTP) on code tokens from the underlying codebase, as well as on file-read observations from agentic trajectories. Consistent with Li et al. (2026), we find that NTP on code tokens (∼650,000 tokens) largely memorizes the codebase while degrading other model capabilities, resulting in performance below the base model. Training on observation tokens is less destructive, but still performs substantially worse than training on distilled note documents. This further underscores the importance of generating notes for acquiring knowledge.

Type II data teaches models to connect internalized knowledge to reasoning and action. We show that Type I notes can internalize substantial environment-specific knowledge, but knowing the environment is not the same as using that knowledge during action. Type II data is designed to provide this missing connection by demonstrating trajectories in which internalized knowledge is recalled, integrated with ongoing reasoning, verified when necessary, and used to select actions. To characterize how this manifests in the final model, we compare C&H trajectories from the base model and the U-WIN model (100M) in Table 1. Environment-specific recall is sparse in the base model, but becomes nearly ubiquitous after world internalization and is routinely interleaved with the model's reasoning. More importantly, this knowledge is not merely surfaced: the model frequently verifies recalled information against the environment and uses it to guide subsequent searches, tool calls, and final submissions. This suggests that internalized knowledge becomes an active component of the model's agentic policy rather than simply information that can be elicited under direct questioning. Figure 5 illustrates this pattern in individual trajectories. In both examples, the trained model begins with a relevant piece of recalled environment knowledge, uses it to immediately narrow its search, verifies the recalled information against the underlying files, and then incorporates it into the final answer. By contrast, the base model follows more generic search strategies and fails to discover the relevant evidence.

Table 1: **% of Recall and knowledge-use behavior** on C&H agentic trajectories.

Metric	Base	U-WIN 100M
Trajectories with clear recall	20.6 ± 1.9	99.8 ± 0.2
Turns containing recall	1.55	40.29
Turns with verification after recall	0.98	57.19
Turns with specific action use	0.07	15.18
Turns with submitted-answer use	0.51	8.03

Type II data also helps preserve agentic behavior. Beyond teaching the model to connect internalized knowledge with reasoning and action, we observe an additional benefit of Type II data in a controlled C&H case study. Figure 6 compares Type I-only models with models trained jointly on Type I and Type II data at matched 30M and 100M training-token budgets. Although Type I-only training improves parametric knowledge and yields modest gains over the base model on the agentic benchmark, joint training produces substantially stronger downstream task performance at both scales. More strikingly, the difference extends beyond final task success: under Type I-only training, failures to follow the agentic interaction protocol become increasingly frequent as training scales,



Figure 5: **Internalized knowledge guides targeted agentic behavior.** Two representative C&H trajectories comparing the base model with U-WIN at 100M. In both cases, U-WIN recalls relevant environment knowledge, uses it to guide targeted search and verification, and completes the task, while the base model follows generic search paths and misses the required evidence.

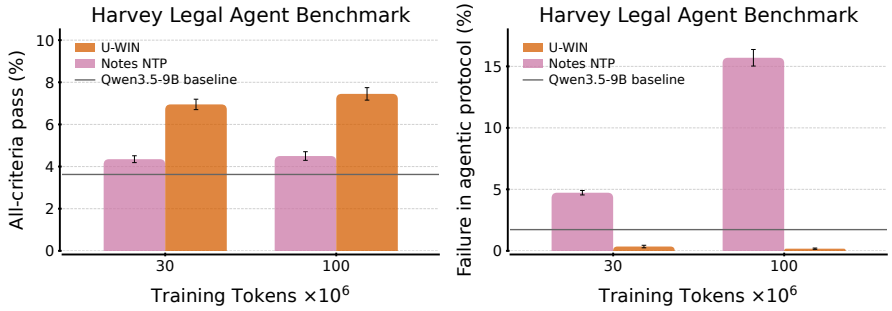


Figure 6: **Type II data also preserves agentic behavior.** Type I-only training (Notes NTP) vs. joint Type I+II training (U-WIN) on C&H at matched 30M and 100M token budgets. **Left:** joint training yields higher agentic task success. **Right:** protocol failures rise sharply with scale under Type I-only training but stay near the base model with joint training. Gray line: Qwen3.5-9B base model.

rising sharply at 100M tokens, whereas joint training keeps such failures close to the base-model level. These results suggest that continued training solely on note-like documents can gradually shift the model away from the behavioral distribution required for agentic interaction, even while improving what the model knows about the environment. Type II trajectories therefore provide a complementary benefit: in addition to demonstrating how internalized knowledge should be incorporated into reasoning and action, they help preserve the tool-use and task-completion behaviors of the post-trained model as knowledge internalization scales.

Conclusion and Limitations. We showed that U-WIN can convert experience from unsupervised exploration into weight updates that improve downstream performance, with gains that grow with the amount of internalized data. This suggests a path toward continual learning, in which a model generates its own training targets from experience. However, we do not run explicit continual or test-time learning experiments: all data comes from the frozen base model, and each environment is internalized once before downstream tasks are posed. Whether gains persist over repeated rounds, sequences of environments, or capabilities outside the target environment remains open.

AI USE STATEMENT

We used generative AI tools to assist with language editing, restructuring and polishing portions of the manuscript, brainstorming titles and terminology, and identifying potentially relevant prior work. We have also utilized coding agents based on generative agent tools for help with running experiments, setting up tasks, and collecting results. We also used generative AI to help organize related-work discussions and suggest citations; though all cited references were subsequently checked by the authors against the corresponding papers or official publication records. AI-assisted text was reviewed, edited, and integrated by the authors, who independently assessed the scientific arguments and claims presented in the paper.

We take responsibility for the final content of this work, including all text, claims, analyses, and artifacts produced with the aid of generative AI.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alex Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=RQm2KQTM5r.

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.1, knowledge storage and extraction. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=5x788rqbcj.

Matthew Bolan, Joachim Breitner, Jose Brox, Nicholas Carlini, Mario Carneiro, Floris van Doorn, Martin Dvorak, Andrés Goens, Aaron Hill, Harald Husum, Hernán Ibarra Mejia, Zoltan A. Kocsis, Bruno Le Floch, Amir Livne Bar-on, Lorenzo Luccioli, Douglas McNeil, Alex Meiburg, Pietro Monticone, Pace P. Nielsen, Emmanuel Osalotioman Osazuwa, Giovanni Paolini, Marco Petracci, Bernhard Reinke, David Renshaw, Marcus Rossel, Cody Roux, Jérémy Scanvic, Shreyas Srinivas, Anand Rao Tadipatri, Terence Tao, Vlad Tsyrklevich, Fernando Vaquerizo-Villar, Daniel Weber, and Fan Zheng. The equational theories project: Advancing collaborative mathematical research at scale, 2025. URL https://arxiv.org/abs/2512.07087.

DatologyAI, :, Pratyush Maini, Vineeth Dorna, Parth Doshi, Aldo Carranza, Fan Pan, Jack Urbanek, Paul Burstein, Alex Fang, Alvin Deng, Amro Abbas, Brett Larsen, Cody Blakeney, Charvi Bannur, Christina Baek, Darren Teh, David Schwab, Haakon Mongstad, Haoli Yin, Josh Wills, Kaleigh Mentzer, Luke Merrick, Ricardo Monti, Rishabh Adiga, Siddharth Joshi, Spandan Das, Zhengping Wang, Bogdan Gaza, Ari Morcos, and Matthew Leavitt. Beyondweb: Lessons from scaling synthetic data for trillion-scale pretraining, 2025. URL https://arxiv.org/abs/2508.10975.

Sabri Eyuboglu, Ryan Ehrlich, Simran Arora, Neel Guha, Dylan Zinsley, Emily Liu, Atri Rudra, James Y Zou, Azalia Mirhoseini, and Christopher Re. Cartridges: Lightweight and general-purpose long context representations via self-study. In *International Conference on Learning Representations*, volume 2026, pp. 42642–42687, 2026.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8342–8360. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.740.

Harvey AI. Harvey lab: The legal agent benchmark, 2026. URL https://github.com/harveyai/harvey-labs/tree/v1.0. Announcement: https://www.harvey.ai/blog/introducing-harveys-legal-agent-benchmark.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-zero: Self-evolving reasoning LLM from zero data. In

The Fourteenth International Conference on Learning Representations, 2026. URL https://openreview.net/forum?id=96apU6YzSO.

Jonas Hübotter, Frederike Lübeck, Lejs Deen Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement learning via self-distillation. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=QkfkxyRizZ.

Haechan Kim, Yoonho Lee, Gisang Lee, Chelsea Finn, and Kangwook Lee. WHALE: A simple recipe for joint harness-weight optimization. *arXiv preprint arXiv:2609.00196*, 2026.

Yoonho Lee, Roshen Sanjay Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-Harness: End-to-end optimization of model harnesses. In *Conference on Language Modeling (COLM)*, 2026. arXiv:2603.28052.

Jacob Xiaochen Li, Rick Battle, and Omar Khattab. Machine studying: A system-level reframing of continual adaptation from declarative corpora. In *Continual Adaptation at Scale: Towards Sustainable AI Workshop*, 2026. URL https://openreview.net/forum?id=e7duOnFgL7.

Bo Liu, Chuanyang Jin, Seungone Kim, Weizhe Yuan, Wenting Zhao, Ilia Kulikov, Xian Li, Sainbayar Sukhbaatar, Jack Lanchantin, and Jason Weston. Spice: Self-play in corpus environments improves reasoning, 2025a. URL https://arxiv.org/abs/2510.24684.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. ProRL: Prolonged reinforcement learning expands reasoning boundaries in large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=YPsJha5HXQ.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Haochen Luo, Yi Huang, Sichun Luo, Fengyuan Liu, Lei Li, Zefa Hu, Junlan Feng, and Qi Liu. Harness-aware self-evolving: Co-evolving model weights, harness, and task solutions. *arXiv preprint arXiv:2607.03935*, 2026.

Pratyush Maini, Skyler Seto, He Bai, David Grangier, Yizhe Zhang, and Navdeep Jaitly. Rephrasing the web: A recipe for compute and data-efficient language modeling. In *ICLR 2024 Workshop on Navigating and Addressing Data Problems for Foundation Models*, 2024. URL https://openreview.net/forum?id=6Gwfc6ZvvW.

Oded Ovadia, Menachem Brief, Moshik Mishaeli, and Oren Elisha. Fine-tuning or retrieval? comparing knowledge injection in LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2024.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Amrith Setlur, Matthew Y. R. Yang, Charlie Victor Snell, Jeremiah Greer, Ian Wu, Virginia Smith, Max Simchowitz, and Aviral Kumar. e3: Learning to explore enables extrapolation of test-time compute for LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=aID0dZmMmM.

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, pp. 8634–8652, 2023.

Vaishnavi Shrivastava, Piero Kauffmann, Ahmed Awadallah, and Dimitris Papailiopoulos. ECHO: Terminal agents learn world models for free. *arXiv preprint arXiv:2605.24517*, 2026.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. AI models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759, July 2024. ISSN 1476-4687. doi: 10.1038/s41586-024-07566-y. URL https://www.nature.com/articles/s41586-024-07566-y.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, Abhishek Kumar, Alex Alemi, Alex Rizkowsky, Azade Nova, Ben Adlam, Bernd Bohnet, Gamaleldin Elsayed, Hanie Sedghi, Igor Mordatch, Isabelle Simpson, Izzeddin Gur, Jasper Snoek, Jeffrey Pennington, Jiri Hron, Kathleen Kenealy, Kevin Swersky, Kshiteej Mahajan, Laura Culp, Lechao Xiao, Maxwell L. Bileschi, Noah Constant, Roman Novak, Rosanne Liu, Tris Warkentin, Yundi Qian, Yamini Bansal, Ethan Dyer, Behnam Neyshabur, Jascha Sohl-Dickstein, and Noah Fiedel. Beyond human data: Scaling self-training for problem-solving with language models. *Transactions on Machine Learning Research*, 2024. arXiv:2312.06585.

Charlie Snell, Dan Klein, and Ruiqi Zhong. Learning by distilling context, 2022. URL https://arxiv.org/abs/2209.15189.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic Cheatsheet: Test-time learning with adaptive memory. *arXiv preprint arXiv:2504.07952*, 2025.

Arnuv Tandon, Karan Dalal, Xinhao Li, Daniel Koceja, Marcel Rød, Sam Buchanan, Xiaolong Wang, Jure Leskovec, Sanmi Koyejo, Tatsunori Hashimoto, Carlos Guestrin, Jed McCaleb, Yejin Choi, and Yu Sun. End-to-end test-time training for long context. *arXiv preprint arXiv:2512.23675*, 2025.

Rishabh Tiwari, Kusha Sareen, Lakshya A. Agrawal, Joseph E. Gonzalez, Matei Zaharia, Kurt Keutzer, Inderjit S. Dhillon, Rishabh Agarwal, and Devvrit Khatri. Learning, fast and slow: Towards LLMs that adapt continually. *arXiv preprint arXiv:2605.12484*, 2026.

Gido M Van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197, 2022.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=NTAhi2JEEE.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. Pmlr, 2022.

Ian Wu, Yuxiao Qu, Amrith Setlur, and Aviral Kumar. Reasoning cache: Continual improvement over long horizons via short-horizon RL. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=IpeH08AJ9P.

Zitong Yang, Neil Band, Shuangping Li, Emmanuel Candès, and Tatsunori Hashimoto. Synthetic continued pretraining. In *International Conference on Learning Representations*, 2025.

Mert Yuksekgonul, Daniel Koceja, Xinhao Li, Federico Bianchi, Jed McCaleb, Xiaolong Wang, Jan Kautz, Yejin Choi, James Zou, Carlos Guestrin, and Yu Sun. Learning to discover at test time. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=96zNuQrH9Y.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 15476–15488, 2022.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic context engineering: Evolving contexts for self-improving language models. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.04618.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Jpxfof0EaS.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Bryan Kian Hsiang Low, and Paul Pu Liang. MEM1: Learning to synergize memory and reasoning for efficient long-horizon agents. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=XY8AaxDSLb.

Deyao Zhu, Xin Zhou, Shengling Qin, Xuekai Zhu, Hangliang Ding, Shu Zhong, et al. Edgebench: Unveiling scaling laws of learning from real-world environments, 2026. URL https://arxiv.org/abs/2607.05155.

Adam Zweiger, Jyothish Pari, Han Guo, Yoon Kim, and Pulkit Agrawal. Self-adapting language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=JsNUE84Hxi.

C&H system prompt for Type II data generation.
Actor prompt. The model is instructed to act as an agent exploring the document-management system of Calderwood & Harkness (C&H), whose files are organized as `matters/<matter-id>/<workstream>/<file>` and exposed as Markdown. It may issue exactly one tool call per turn using `glob`, `grep`, or `read`, receives the result in a `<tool_result>` block, and must finish with a `<final>` answer. Final answers must cite supporting file paths and may not fabricate document contents.
Memory instruction. For Type II generation, the teacher additionally receives a block of remembered passages, described as prior knowledge the model already remembers. The prompt instructs the teacher to use this remembered knowledge when relevant, distinguish recalled information from newly observed evidence, resolve uncertainty with tools when needed, and write naturally without referring to the supplied notes themselves.

Figure 7: **C&H system prompt used to generate Type II data.** The teacher is given the standard C&H agent prompt together with an additional memory instruction that presents selected notes as remembered prior knowledge.

A Appendix

A.1 Training Details

Data generation.

For C&H and Equational Theories, both the student and data generator use the same frozen Qwen3.5-9B checkpoint (revision `c202236`); no stronger teacher is used. Type I data consists of notes written during autonomous exploration. C&H collects notes over a seed round followed by eight recursive rounds, where later exploration can search and build on previously written notes. Equational Theories similarly uses recursive rounds, with access to the existing note bank as well as Lean and finite-table checkers during exploration. Raw exploration trajectories are not directly used as Type I training data. :chatgpt-content-referenceindex="0"

Type II data consists of note-conditioned solutions generated by the frozen base model. The teacher receives relevant notes as privileged context, while the student is trained on the resulting trajectory with this note context removed. For C&H, Type II examples are multi-turn tool-use trajectories with loss on assistant reasoning, tool calls, and final answers; tool observations are masked. For Equational Theories, Type II examples are single-turn mathematical responses, with only the final response retained for student training. Neither environment filters Type II trajectories by task correctness. :chatgpt-content-referenceindex="1"

Training. Unless otherwise specified, we mix Type I and Type II data at a 70/30 ratio of supervised tokens. Training sets at different scales are nested subsets of a fixed ordering, so smaller-scale runs are contained within larger ones. We train for two epochs with learning rate 5×10^{-6}, cosine decay, 3% warmup, bf16 precision, FSDP2, and activation checkpointing. C&H uses packed sequences of 16,384 tokens and four rows per update; Equational Theories uses 17,408-token packed sequences and eight rows per update with per-example attention isolation. The main runs use full-parameter fine-tuning without replay, KL regularization, or model merging; we study KL regularization separately as an ablation. :chatgpt-content-referenceindex="2"

A.2 Environment Details

C&H. C&H is the document-management system of the fictional law firm Calderwood & Harkness LLP from the Harvey Legal Agent Benchmark. The environment contains 9,288 documents across 266 matters and 46 clients. Documents are converted to Markdown and exposed through three read-only tools: `glob`, `grep`, and `read`. During unsupervised exploration, the model operates for up to 20 turns with thinking enabled; later recursive rounds additionally provide note-search and note-writing tools. :chatgpt-content-referenceindex="3"

For closed-book evaluation, we use 1,780 multiple-choice probes selected from a larger suite of environment-specific facts extracted from and verified against the C&H documents. Each probe contains one true statement and three same-relation distractors. The model receives no tools or chat template; each candidate is scored by its summed sequence log-probability, and accuracy is

Table 2: Training details for C&H and Equational Theories. Token budgets refer to supervised, loss-bearing tokens per epoch.

Setting	C&H	Equational Theories
Base / generator	Qwen3.5-9B	Qwen3.5-9B
Type I data	Recursive note bank	Recursive note bank
Type II data	Multi-turn tool-use trajectories	Single-turn mathematical responses
Type I / II mix	70 / 30	70 / 30 (100M: 61 / 39)
Training scales	1, 5, 10, 30, 100M	1, 3, 10, 30, 100M
Sequence length	16,384	17,408
Batch	4 rows / update	8 rows / update
Optimization	2 epochs, LR 5×10^{-6}, cosine, 3% warmup, bf16	
Main-run preservation	None	None

Table 3: Evaluation protocols for C&H and Equational Theories.

Evaluation	Size	Access at evaluation	Metric
C&H closed-book	1,780 MC probes	None	MC accuracy
C&H agentic	250 tasks $\times$ 16 samples	`glob`/`grep`/`read`	All-criteria pass
Equational Stage 1	800 questions $\times$ 4 samples	None	Verdict accuracy
Equational direct recall	424 facts $\times$ 2 forms	None	True/False accuracy

reported over probes. Because these probes are derived from information encountered during exploration, they measure retention of environment knowledge rather than generalization to unseen facts. :chatgpt-content-referenceindex="4"

For downstream agentic evaluation, we use the benchmark's 250 firm-knowledge tasks. The model receives `glob`, `grep`, and `read`, a 20-turn budget, and no access to exploration notes. Each task is graded against its full rubric, and an attempt succeeds only if all criteria pass. In the final evaluation we draw 16 samples per task and report all-criteria pass rate. :chatgpt-content-referenceindex="5"

Equational Theories. The Equational Theories environment contains 4,694 magma laws together with known implications, dualities, Lean proofs, counterexamples, finite tables, and method descriptions. During exploration, the model can inspect this bank, read and edit notes, and use sandboxed Lean and finite-table checkers. Later rounds are seeded from gaps or conflicts in the existing knowledge bank, and the model must first formulate its own question before writing new notes. :chatgpt-content-referenceindex="6"

The main downstream evaluation asks whether one law implies another over all magmas. We use four 200-example splits—normal, hard, extra-hard, and order-5—each balanced between true and false implications. The model produces four samples per problem with thinking enabled; only the final True/False verdict is scored. No notes, retrieval, tools, or environment access are available at evaluation time. Exact evaluation pairs are removed from the study bank and blocked during exploration and Type II construction, although transfer through equivalent or dual laws is allowed. :chatgpt-content-referenceindex="7"

A.3 Direct Recall Evaluation on Equational Theories

We additionally evaluate whether knowledge written into the model's training notes can later be recovered directly from its parameters. Unlike the main Equational Theories evaluation, which tests held-out implication relationships and allows long-form reasoning, this evaluation consists of short True/False questions about facts contained in the training notes. Thinking is disabled, decoding is greedy, and the model receives no tools, notes, or environment access. :chatgpt-content-referenceindex="8"

The evaluation contains 424 facts across three categories. *Implications* test whether one law implies another; *collapse lemmas* test whether a law forces the underlying magma to be trivial; and *countermodel properties* test whether a law holds on a specified two-element magma. Each fact is asked in two forms: one using the catalogue identifier and one formula-only version with renamed variables, yielding 848 prompts in total. Labels are balanced within each category. :chatgpt-content-referenceindex="9"

A response is correct only if it gives the correct True/False label. Overall accuracy is the micro-average over all prompts. Exact downstream SAIR evaluation pairs are excluded, but the recall facts are intentionally drawn from note content used for training, since this evaluation measures retention rather than generalization. One caveat is that smaller-scale models have not necessarily seen every fact included in the fixed evaluation set, and some items may also be solvable through short reasoning rather than pure memorization. :chatgpt-content-referenceindex="10"

Table 4: The Equational Theories

Model	**Overall**	**Implications**	**Collapse Lemmas**	**Countermodel Properties**
Base	51.65%	52.84%	54.04%	49.50%
1M	50.35%	51.70%	50.00%	50.00%
3M	53.77%	64.20%	52.21%	50.25%
10M	58.73%	77.84%	55.88%	52.25%
30M	60.14%	79.55%	58.46%	52.75%
100M	**66.86%**	**79.55%**	**73.16%**	**57.00%**