HETEROGENEOUS AGENTIC COEVOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advances in self-improving LLM agents have demonstrated the potential of AI systems that drive their own intelligence. However, many domain-specific applications rely on *specialized foundation models* that possess strong domain expertise but remain dependent on human-designed improvements. Extending recursive self-improvement to these models presents a key challenge: as a specialist improves, the strategies used to advance it must also adapt. To address this challenge, we propose **Heterogeneous Agentic Coevolution (HACE)**, a framework that jointly evolves *specialized foundation models* and *LLM agents* through reciprocal learning. HACE improves the specialist through supervised learning while training the agent with on-policy reinforced self-training, guided by verified performance gains on the current specialist. The agent's exploration, in turn, informs subsequent specialist training, enabling both models to adapt to each other's progress. To sustain this reciprocal learning, HACE further employs evolutionary search to optimize their interaction harness, adapting how the models exchange information and select training experiences as their capabilities evolve. We evaluate HACE across three challenging domains: recommendation systems, protein function prediction in biology, and crystal stability prediction in material discovery. We show that HACE outperforms the evaluated self-improving methods on the primary task metrics across all three domains. These results shed light on heterogeneous coevolution as a universal approach to improving specialized foundation models through reciprocal interaction with a learning agent.

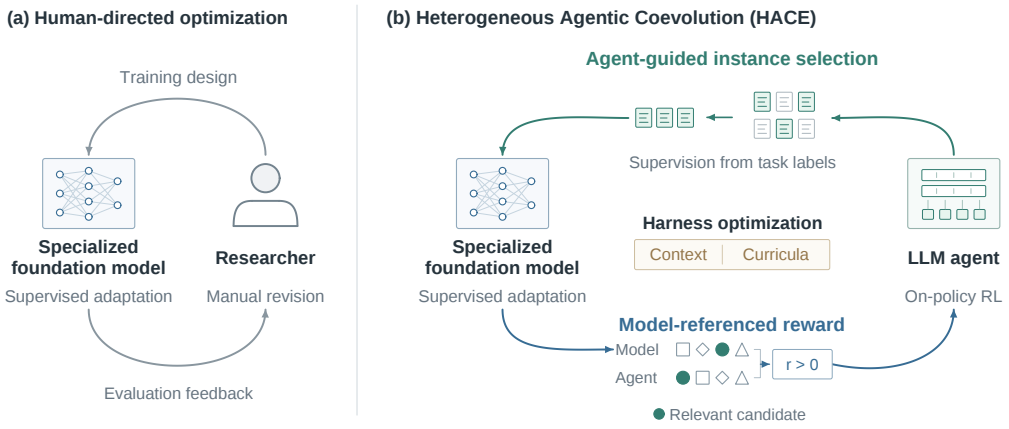

Figure 1: From human-directed optimization to heterogeneous agentic coevolution. HACE brings together *specialized foundation models* and *LLM agents* into a shared learning process, where each helps determine how the other improves.

1 INTRODUCTION

Lately, Large Language Models (LLM) agents are beginning to automate the experimental work behind model development. Agents can revise their implementations, improve their interaction harnesses, and modify the procedures that generate further improvements (Zhang et al., 2026a; Lee et al., 2026; Zhang et al., 2026b). They can also continue training on experience collected during

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.

adaptation and exploration (Zweiger et al., 2025; Yuksekgonul et al., 2026). These advances suggest that model development itself can become an autonomous learning process. Extending this progress to scientific AI requires bringing the domain expertise of specialized foundation models into this process.

In scientific AI, specialized foundation models generally learn reusable domain knowledge through large-scale pretraining. Protein language models learn biological regularities from sequences (Lin et al., 2023), while pretrained interatomic potential models learn the energies and forces governing material behavior (Deng et al., 2023). In recommendation, architectures such as HSTU (Zhai et al., 2024) and Wukong (Zhang et al., 2024) extend this scaling approach to large collections of user interactions. These models acquire domain expertise that often *exceeds* that of general LLM agents due to specific designs. Yet they still cannot self-evolve like LLM agents. Once supervised training reaches convergence on test distributions with no improvements, researchers have to manually identify useful designs to improve the specialized foundation model, such as better feature engineering or algorithmic changes, as shown in Figure 1.

Recent developments in agentic systems pave a feasible path to automate this process. Their ability to interpret feedback, explore alternatives, and revise executable procedures allows them to influence how another model learns. For example, Eureka (Ma et al., 2024) demonstrates this possibility by using LLMs to design reward programs for training robot policies autonomously. However, most self-evolving systems still rely on proprietary LLMs whose parameters remain fixed while agent code and interaction harnesses evolve (Zhang et al., 2026a; Lee et al., 2026; Zhang et al., 2026b). The experience collected during improvement thus does not update the underlying language model. For specialized foundation models, the challenge is for the LLM agent model to keep improving the specialized model as the specialized model advances. An intervention that improves the initial model can become redundant after further training. An agent that learns to address the initial model's weaknesses, therefore, faces a changing task as those weaknesses are resolved. Continued progress requires the agent to learn continually with the model when it improves. We refer this mechanism as *reciprocal learning*.

To close the gap, we propose a novel framework, **Heterogeneous Agentic Coevolution (HACE)**, which coevolves a specialized foundation model (i.e., the specialist) with a heterogeneous LLM agent which learns to guide the coevolution. The key idea is to improve the specialist through supervision while also training the agent to keep advancing it under the certain curriculum. Agent learning is guided by verified gains over the current specialist. The agent's exploration, in turn, guides the selection of experiences for subsequent specialist training. As the specialist improves, it changes which agent contributions are useful. As the agent improves, it changes the learning opportunities available to the specialist. This reciprocal feedback loop seamlessly connects supervised specialist learning with on-policy reinforced self-training.

We need to ensure these models to interact and develop their capabilities in a continuously improving direction. What the agent sees and which experiences reach specialist training determine the opportunities that either model can use. Thus, we also propose an evolutionary interaction harness in HACE that governs information exchange and training experience selection. Evolutionary search evaluates harness changes through their effects on task performance and specialist learning. A better harness can expose opportunities that the current models have not utilized, while model updates change which harness designs remain effective. HACE thus improves the specialist while learning how to organize its continued improvement. Domain expertise, agentic capabilities, and the process connecting their learning coevolve together.

To show the effectiveness, we comprehensively evaluate HACE in three challenging domains, including recommendation systems, protein function prediction in biology, and crystal stability prediction in material discovery. In each domain, we begin with specialists already *trained to convergence*, where continued training under the original supervised procedure yields no further improvement. This setting stress-tests whether interaction with an evolving agent can create opportunities for learning beyond that plateau. Across diverse specialist architectures and large-scale datasets, HACE achieves state-of-the-art performance and outperforms all evaluated evolution methods. These results demonstrate that HACE enables specialists to continue improving beyond supervised convergence. HACE brings their existing domain expertise into autonomous self-improvement while developing the agentic capabilities needed to advance it.

To summarize, our main contributions are as follows:

- We introduce HACE, an evolutionary framework for autonomous self-improvement that brings specialized foundation models and LLM agents into a shared coevolution process.
- We propose a novel reciprocal learning mechanism in which verified agent contributions guide agent learning, agent exploration also informs specialist training, and an evolving harness adaptively coordinates their reciprocal interaction.
- We empirically demonstrate the effectiveness of HACE across three challenging domains: recommendation, protein function prediction, and material discovery. The results show that HACE can substantially push the domain-specific performance beyond the capability boundary.

2 Related Work

Agent and harness optimization. Recent work uses execution feedback to improve agent programs, interaction harnesses, and the procedures that propose further improvements (Hu et al., 2025; Zhang et al., 2026a; Lee et al., 2026; Luo et al., 2026; Zhang et al., 2026b; Cemri et al., 2026; Liu et al., 2026b). Other methods improve the underlying model through parameter updates. Reinforced self-training learns from responses selected by task rewards (Singh et al., 2024), while self-adaptation and test-time learning use generated experience or task feedback for further training (Zweiger et al., 2025; Yuksekgonul et al., 2026). HACE connects harness search with parameter learning in both a specialized foundation model and an LLM agent. The harness controls their interaction and the experience used to train each model.

Coevolution and reciprocal learning. Coevolution can be traced to reciprocal evolutionary responses between interacting populations in biology (Janzen, 1980). In machine learning, many methods have connected the development of reward programs, environments, or optimization procedures with the learners they guide (Ma et al., 2024; Liu et al., 2026c; Wang et al., 2019; Dennis et al., 2020; Chen et al., 2026; Yang et al., 2026; Li et al., 2026). HACE studies this dependence between models with different capabilities and learning signals. Agent exploration can change which instances train the specialized model, whose predictions in turn determine which agent responses pass verification. Our analysis characterizes this dependence through the distribution of verified experience and identifies when an update to either learner changes its allocation across instances.

Agent distillation. Agent distillation transfers reasoning and tool use from LLM agents to smaller language models (Liu et al., 2026a; Kang et al., 2025). MAD-OPD derives supervision from teachers that debate over the student's current trajectories (Wang et al., 2026). These methods update the student while keeping teacher parameters fixed. HACE is *distinct* from this line of works and takes a step forward, where an LLM agent with lower overall domain performance can *still* improve on the specialized foundation model. Agent exploration guides instance selection for specialist training, while verified contributions guide response selection back for agent learning. Both models can therefore coevolve reciprocally without assigning either a fixed teacher role.

3 Heterogeneous Agentic Coevolution

In this section, we formally present Heterogeneous Agentic Coevolution (HACE), a framework for reciprocal self-improvement between specialized foundation models and LLM agents. Figure 2 illustrates how an evolving harness connects specialist adaptation, joint selection, and agent learning within one coevolution loop. In Section 3.1, we describe how specialist predictions can guide agent exploration and how the resulting experience selects instances for specialist training with original task labels. In Section 3.2, we incorporate the proposed harness together with the adapted specialist, then use the retained pair to identify fresh responses for on-policy agent learning. These updates allow each learner to shape the training experience of the other precisely. In Section 3.3, we establish a population risk bound that relates specialist learning to the coverage of accepted experience within the coevolution.

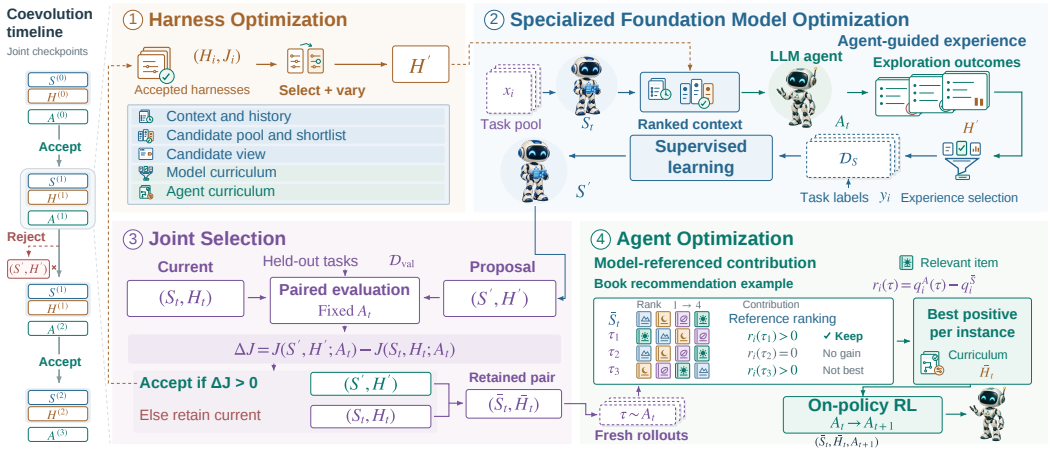

Figure 2: Overview of HACE. Agent exploration selects specialist training experience. Joint validation retains the specialist and harness, after which fresh verified responses train the agent and shape later exploration. The on-policy agent optimization step uses reinforced self-training.

3.1 Reciprocal Learning

Let S denote the specialist, A the agent, and H the harness. Round $t = 0, 1, \ldots$ starts from (S_t, H_t, A_t). We propose a harness H' by modifying a parent from an archive of accepted harnesses. Its fields control the task information and specialist context shown to the agent, how responses are interpreted, and which examples train each learner. Let $\mathcal{I}$ index the finite nonempty training pool of labeled instances (x_i, y_i).

Under H', the agent receives a prompt constructed from a training input x_i and the predictions of S_t, without access to its label y_i. We evaluate each response τ against y_i using the same task constraints as the specialist. Let $q_i^A(\tau)$ and q_i^S denote their respective task qualities, with larger values preferred. The response contribution is

$$r_i(\tau) = q_i^A(\tau) - q_i^S. \tag{1}$$

A response passes verification when its prediction is valid, supervision is evaluable, and $r_i(\tau) > 0$. An agent with lower average task quality can still provide verified responses on individual instances. The specialist and harness used in each collection determine both quality terms. Appendix B.3 specifies the task quality and verification rules.

The harness also determines which prediction candidates the agent can access. Additional sampling cannot recover a relevant candidate excluded from that view. Changing the view can therefore create opportunities that further sampling under the same harness would miss, while the curriculum determines which exploration outcomes receive the specialist's training budget.

The harness uses these outcomes to select a nonempty set of training indices $\mathcal{I}_t^S \subseteq \mathcal{I}$. Its curriculum can prioritize verified improvements, unsuccessful exploration, or uncertain joint predictions, with uniform sampling as an option. Write S_ϕ for the specialist with trainable parameters ϕ, which may belong to the foundation model or its prediction head. A fixed optimization budget starting from S_t produces S' using

$$\mathcal{L}_{S,t}(\phi) = \frac{1}{|\mathcal{I}_t^S|} \sum_{i \in \mathcal{I}_t^S} \ell_{\text{task}}(S_\phi(x_i), y_i), \tag{2}$$

where ℓ_{task} is the original supervised task loss. Agent exploration determines where to train, while task labels determine what the specialist learns. The harness thus affects both current predictions and the specialist produced by adaptation. We evaluate H' and S' **together after specialist adaptation** to account for both effects.

3.2 Joint Selection

Joint selection tests whether specialist adaptation under the harness improves joint predictions. For a candidate c in the view for instance i, let $s_i(c)$ be its specialist score after any stored calibration,

with larger values preferred, and $a_i(c)$ its agent score. Joint prediction ranks candidates by

$$s_i(c) + \lambda a_i(c), \qquad \lambda \geq 0. \tag{3}$$

Here $a_i(c) = 1/\log_2(u+1)$ when c appears at position u in the agent ranking, and zero otherwise. The weight λ controls the agent's influence, with zero recovering the specialist ordering within the same view.

Training, validation, and final evaluation use disjoint splits. With A_t fixed, we compare the proposed and current pairs on the same validation batch $\mathcal{B}_t$. Let $J_{\mathcal{B}_t}(S, H, A)$ be the mean joint task quality for state (S, H, A). Both states use the same two folds, fitting their own weights from a fixed grid on one fold and evaluating on the other, then exchanging roles. The proposal is accepted when

$$\Delta J_t = J_{\mathcal{B}_t}(S', H', A_t) - J_{\mathcal{B}_t}(S_t, H_t, A_t) > 0. \tag{4}$$

Acceptance retains (S', H') and records the harness and score. Otherwise, we restore (S_t, H_t) and the specialist training state. Write $(\bar{S}_t, \bar{H}_t)$ for the pair retained after this decision.

The retained pair now defines the context and reference for agent learning. A specialist update can change which responses improve on its predictions, and a harness update can change the information available to the agent. We therefore collect fresh responses with A_t under $(\bar{S}_t, \bar{H}_t)$, keep the original response with the largest positive contribution for each eligible instance, and apply the agent curriculum. This makes agent supervision depend on the current specialist's remaining opportunities for improvement.

Let $\mathcal{Z}_t$ be the indexed dataset of selected prompt and response concatenations, with repeated records counted separately. For a record z, let $|z|$ count its nonpadding tokens, z_ℓ be its token at position ℓ, and $z_{<\ell}$ its preceding prefix. Write $p_\theta(z_\ell \mid z_{<\ell})$ for the agent's next-token probability before decoding transformations, with trainable parameters θ. For nonempty $\mathcal{Z}_t$, on-policy reinforced self-training (Singh et al., 2024) minimizes

$$\mathcal{L}_{A,t}(\theta) = -\frac{1}{|\mathcal{Z}_t|} \sum_{z \in \mathcal{Z}_t} \frac{1}{|z| - 1} \sum_{\ell=2}^{|z|} \log p_\theta(z_\ell \mid z_{<\ell}). \tag{5}$$

Each record has at least two tokens. The loss averages predicted tokens within each record, including prompt and response tokens, then averages records. Contribution scores select training responses without weighting this loss. The dataset stays fixed during fitting, and an empty dataset leaves the agent unchanged. Agent learning follows either selection outcome, giving the next state $(S_{t+1}, H_{t+1}, A_{t+1}) = (\bar{S}_t, \bar{H}_t, A_{t+1})$. Its updated responses guide exploration in the next round.

3.3 Learning Guarantee

The training distribution can change with model updates and harness selection. To study learning under that changing distribution, we theoretically analyze on accepted exposure and the final specialist's loss on the resulting experience. Let $\mu_i \geq 0$, with $\sum_{i \in \mathcal{I}} \mu_i = 1$, be fixed source probabilities. Fix a finite nonempty specialist class $\mathcal{F}$ before exploration and a fixed loss $\ell_i(f) \in [0, 1]$ for each $i \in \mathcal{I}$ and $f \in \mathcal{F}$. This analysis loss is independent of the harness and may differ from ℓ_{task}. The specialist's source risk is $R_\mu(f) = \sum_{i \in \mathcal{I}} \mu_i \ell_i(f)$.

Let $\mathcal{H}_t$ be the observed history before round t, and let $W_{t,i}$ be the accepted training weight of instance i. A nonempty accepted update assigns total weight one uniformly to newly selected instances. All other weights are zero. These weights are known by the end of the round. After T rounds, define cumulative expected accepted exposure and loss by

$$C_T(i) = \sum_{t<T} \mathbb{E}[W_{t,i} \mid \mathcal{H}_t], \qquad \widehat{L}_T(f) = \sum_{t<T} \sum_{i \in \mathcal{I}} W_{t,i} \ell_i(f), \tag{6}$$

where $t < T$ denotes rounds $0, \ldots, T-1$. In $\widehat{L}_T(S_T)$, all accepted instances are evaluated at the final specialist $S_T \in \mathcal{F}$, which may depend on the observed history. For $v > 0$, let $U_T(v) = \sum_{i: C_T(i) < v\mu_i} \mu_i$ be the source mass with insufficient exposure.

Theorem 1 (Learning from Evolving Experience). *For any fixed $\delta \in (0, 1)$, with probability at least $1 - \delta$ over the learning process, every final specialist $S_T \in \mathcal{F}$ satisfies, simultaneously for all*

integers $T \geq 1$ *and all* $v > 0$*,*

$$R_\mu(S_T) \leq U_T(v) + \frac{\widehat{L}_T(S_T) + \log(|\mathcal{F}|/\delta)}{(1 - e^{-1})v}. \tag{7}$$

On the same event, $R_\mu(S_T) \to 0$ *whenever* $v_T > 0$ *satisfies* $v_T \to \infty$*,* $U_T(v_T) \to 0$*, and* $\widehat{L}_T(S_T)/v_T \to 0$*.*

The bound separates reaching useful training instances from learning and retaining their information. Harness search, agent exploration, and joint selection determine accepted exposure. Specialist optimization determines the final loss on that experience, including whether later updates preserve earlier learning. Sufficient exposure across the source distribution and small loss relative to that exposure yield a small risk bound, even when selection depends on the full preceding history. Appendix B.1 gives the complete proof. Proposition 1 explains how agent fitting and reference changes affect verification before curriculum selection and acceptance determine exposure.

4 Experiments

In this section, we empirically evaluate whether HACE can push the capability boundary with the converged specialized foundation models across three important domains (recommendation systems, protein function prediction, and material discovery), compared with other self-improving methods.

Experimental setups. By default, we use Gemma4-E4B (Gemma Team, 2026) as the LLM agent. We also conduct more experiments with various LLM agents (Gemma4-E2B, Gemma4-26B-A4B) in Section 4.3. On the specialized foundation models, for recommendation tasks, we evaluate on MovieLens 1M (ML-1M) (Harper & Konstan, 2015) and four Amazon categories (McAuley et al., 2015) using three mainstream foundation models SASRec (Kang & McAuley, 2018), HSTU (Zhai et al., 2024), and Wukong (Zhang et al., 2024). For protein function prediction, we use frozen ESM-2 (Lin et al., 2023) and ProtT5-XL (Elnaggar et al., 2022) backbones on CAFA (Jiang et al., 2016). For crystal stability prediction in material discovery task, we use CHGNet (Deng et al., 2023) on WBM (Wang et al., 2021; Riebesell et al., 2025). Base denotes the initial specialists, which are already trained to convergence before coevolution. Unless stated otherwise, reported scores apply the fusion in Equation 3 with the weight selected by cross fitting, for Base and for all methods alike. At round 0 the agent is untrained and cross fitting selects a zero weight. Table 7 shows that HACE is agnostic to the specialist architectures and their adaptation scope, including causal attention models, a factorization network, protein language models, and a graph neural potential model.

We compare against popular self-improving methods, including ReST-EM (Singh et al., 2024), Meta-Harness (Lee et al., 2026), OpenEvolve (Sharma, 2025), DGM (Zhang et al., 2026a), and TTT-Discover (Yuksekgonul et al., 2026). The recommendation task performs ranking on candidate items, the protein prediction task predicts the function terms, and the materials discovery task predicts crystal structures within a defined chemical system. These experiments measure improvements in prediction with their pretrained representations.

Evaluation metrics. We use Hit Rate (HR@10) measures the fraction of recommendation queries with the target item among the first ten predictions. NDCG is calculated by binary relevance with discounting each relevant candidate at rank r by $1/\log_2(r+1)$, and normalizes by the ideal ranking within the same reference pool. We average NDCG@5 and NDCG@10 on CAFA, and NDCG@3 and NDCG@5 on WBM according to the baseline implementations. For protein function prediction, $F_{\max}$ is the maximum harmonic mean of precision and recall over prediction thresholds (Jiang et al., 2016). For material discovery, we use F_1 to measure their harmonic mean for identifying stable structures (Riebesell et al., 2025). Higher values indicate better performance for these metrics. Appendix C.1 defines all metrics in detail, including the additional measures reported there. Tables report means with standard deviations on three random seeds. Joint selection and fusion calibration only use the split validation data to avoid data contamination, as defined in Section 3.2.

4.1 Main Comparison

Recommendation. HACE achieves the highest mean HR@10 in all 15 settings in Table 1. Relative gains over Base range from 2.9% to 18.5%, averaging 12.1% across settings. On ML-1M, sev-

Table 1: Main results on recommendation across five datasets and three architectures. Entries report HR@10 ↑ with standard deviations in subscript. **Bold** marks the best mean in each row.

Dataset	Base	ReST-EM	Meta-Harness	OpenEvolve	DGM	TTT-Discover	**HACE**
SASRec							
ML-1M	$0.2557_{\pm.0012}$	$0.2492_{\pm.0022}$	$0.2491_{\pm.0017}$	$0.2421_{\pm.0004}$	$0.2488_{\pm.0018}$	$0.2473_{\pm.0013}$	$\mathbf{0.2632}_{\pm.0013}$
Beauty	$0.0599_{\pm.0004}$	$0.0597_{\pm.0010}$	$0.0631_{\pm.0007}$	$0.0638_{\pm.0003}$	$0.0634_{\pm.0005}$	$0.0633_{\pm.0004}$	$\mathbf{0.0672}_{\pm.0004}$
CDs	$0.0576_{\pm.0005}$	$0.0587_{\pm.0010}$	$0.0604_{\pm.0004}$	$0.0611_{\pm.0002}$	$0.0592_{\pm.0008}$	$0.0587_{\pm.0007}$	$\mathbf{0.0647}_{\pm.0006}$
Sports	$0.0315_{\pm.0003}$	$0.0325_{\pm.0001}$	$0.0323_{\pm.0002}$	$0.0331_{\pm.0004}$	$0.0333_{\pm.0003}$	$0.0323_{\pm.0002}$	$\mathbf{0.0356}_{\pm.0003}$
Toys	$0.0596_{\pm.0001}$	$0.0648_{\pm.0007}$	$0.0657_{\pm.0008}$	$0.0662_{\pm.0008}$	$0.0661_{\pm.0009}$	$0.0646_{\pm.0008}$	$\mathbf{0.0706}_{\pm.0003}$
HSTU							
ML-1M	$0.2604_{\pm.0003}$	$0.2563_{\pm.0005}$	$0.2569_{\pm.0007}$	$0.2573_{\pm.0005}$	$0.2572_{\pm.0006}$	$0.2588_{\pm.0013}$	$\mathbf{0.2685}_{\pm.0003}$
Beauty	$0.0626_{\pm.0003}$	$0.0669_{\pm.0013}$	$0.0673_{\pm.0013}$	$0.0668_{\pm.0013}$	$0.0687_{\pm.0004}$	$0.0695_{\pm.0003}$	$\mathbf{0.0731}_{\pm.0003}$
CDs	$0.0657_{\pm.0002}$	$0.0654_{\pm.0006}$	$0.0656_{\pm.0008}$	$0.0656_{\pm.0010}$	$0.0658_{\pm.0006}$	$0.0673_{\pm.0003}$	$\mathbf{0.0706}_{\pm.0002}$
Sports	$0.0348_{\pm.0003}$	$0.0346_{\pm.0005}$	$0.0349_{\pm.0004}$	$0.0351_{\pm.0003}$	$0.0372_{\pm.0001}$	$0.0372_{\pm.0002}$	$\mathbf{0.0394}_{\pm.0001}$
Toys	$0.0673_{\pm.0001}$	$0.0727_{\pm.0006}$	$0.0732_{\pm.0003}$	$0.0726_{\pm.0002}$	$0.0746_{\pm.0007}$	$0.0740_{\pm.0004}$	$\mathbf{0.0788}_{\pm.0004}$
Wukong							
ML-1M	$0.2559_{\pm.0020}$	$0.2612_{\pm.0008}$	$0.2573_{\pm.0013}$	$0.2582_{\pm.0003}$	$0.2610_{\pm.0013}$	$0.2607_{\pm.0013}$	$\mathbf{0.2703}_{\pm.0020}$
Beauty	$0.0734_{\pm.0002}$	$0.0810_{\pm.0004}$	$0.0805_{\pm.0004}$	$0.0795_{\pm.0004}$	$0.0788_{\pm.0004}$	$0.0772_{\pm.0004}$	$\mathbf{0.0856}_{\pm.0002}$
CDs	$0.0726_{\pm.0005}$	$0.0771_{\pm.0002}$	$0.0798_{\pm.0004}$	$0.0780_{\pm.0004}$	$0.0785_{\pm.0004}$	$0.0770_{\pm.0004}$	$\mathbf{0.0842}_{\pm.0005}$
Sports	$0.0388_{\pm.0005}$	$0.0408_{\pm.0002}$	$0.0398_{\pm.0002}$	$0.0390_{\pm.0002}$	$0.0410_{\pm.0002}$	$0.0395_{\pm.0002}$	$\mathbf{0.0438}_{\pm.0007}$
Toys	$0.0759_{\pm.0002}$	$0.0785_{\pm.0004}$	$0.0752_{\pm.0004}$	$0.0793_{\pm.0004}$	$0.0820_{\pm.0004}$	$0.0770_{\pm.0004}$	$\mathbf{0.0866}_{\pm.0006}$

Table 2: Main results on protein function prediction (CAFA) and material discovery (WBM). Higher is better for all metrics. Full scientific metrics appear in Tables 5 and 6.

	CAFA / ESM-2		CAFA / ProtT5-XL		WBM / CHGNet	
Method	NDCG ↑	$F_{\max}$ ↑	NDCG ↑	$F_{\max}$ ↑	NDCG ↑	F_1 ↑
Base	$0.6285_{\pm.0019}$	$0.2218_{\pm.0014}$	$0.6241_{\pm.0011}$	$0.2221_{\pm.0005}$	$0.6388_{\pm.0011}$	$0.5159_{\pm.0003}$
ReST-EM	$0.6286_{\pm.0018}$	$0.2219_{\pm.0014}$	$0.6242_{\pm.0011}$	$0.2221_{\pm.0005}$	$0.6496_{\pm.0006}$	$0.5508_{\pm.0004}$
Meta-Harness	$0.6309_{\pm.0015}$	$0.2224_{\pm.0013}$	$0.6261_{\pm.0017}$	$0.2229_{\pm.0006}$	$0.6513_{\pm.0008}$	$0.5510_{\pm.0004}$
OpenEvolve	$0.6320_{\pm.0018}$	$0.2230_{\pm.0012}$	$0.6254_{\pm.0014}$	$0.2230_{\pm.0003}$	$0.6486_{\pm.0008}$	$0.5463_{\pm.0003}$
DGM	$0.6325_{\pm.0019}$	$0.2219_{\pm.0014}$	$0.6273_{\pm.0016}$	$0.2235_{\pm.0006}$	$0.6496_{\pm.0007}$	$0.5508_{\pm.0003}$
TTT-Discover	$0.6287_{\pm.0018}$	$0.2219_{\pm.0014}$	$0.6241_{\pm.0011}$	$0.2221_{\pm.0005}$	$0.6474_{\pm.0007}$	$0.5504_{\pm.0003}$
HACE	$\mathbf{0.6745}_{\pm.0016}$	$\mathbf{0.2438}_{\pm.0019}$	$\mathbf{0.6739}_{\pm.0013}$	$\mathbf{0.2317}_{\pm.0005}$	$\mathbf{0.6541}_{\pm.0010}$	$\mathbf{0.5513}_{\pm.0004}$

eral evolution methods fall below the initial SASRec and HSTU models, while HACE improves all three architectures. On the four Amazon categories, Wukong has the strongest initial scores among the three architectures, yet HACE further improves its HR@10 by 12.9% to 16.6%. Contribution rewards train the agent against the current specialist throughout this process.

Protein function prediction. Table 2 shows improvements with both protein backbones. HACE raises NDCG from 0.6285 to 0.6745 with ESM-2 and from 0.6241 to 0.6739 with ProtT5-XL. Existing evolution methods improve over Base by at most 0.0040 and 0.0032, respectively, compared with 0.0460 and 0.0498 for HACE. HACE also leads in $F_{\max}$, AUPR, and Recall@10. These gains cover both the ordering of function terms and their classification. Table 5 also reports semantic distance, for which DGM has the lowest mean on ProtT5-XL. The ablations below examine how these gains depend on harness search and specialist optimization.

Material Discovery. On WBM, all competing methods improve NDCG over Base, from 0.6388 to at least 0.6474. HACE reaches 0.6541, exceeding the strongest competing mean by 0.0028. HACE also achieves the best mean on all four additional metrics in Table 6. Removing specialist optimization reduces NDCG to 0.6395, close to Base at 0.6388. The contribution of supervised specialist updates is therefore particularly clear on this task. Supervised adaptation improves numerical predictions, while the ranking objective favors recovering stable candidates near the top.

Table 3: Ablation studies on recommendation (Toys/SASRec), protein function prediction (CAFA/ESM-2 650M), and material discovery (WBM/CHGNet). Metrics are HR@10 for Toys and mean NDCG for CAFA and WBM.

Leave-one-out component	Toys	CAFA	WBM
HACE	$\mathbf{0.0706}_{\pm.0003}$	$\mathbf{0.6745}_{\pm.0016}$	$\mathbf{0.6541}_{\pm.0010}$
w/o harness optimization	$0.0658_{\pm.0005}$	$0.6307_{\pm.0019}$	$0.6500_{\pm.0012}$
w/o specialist optimization	$0.0641_{\pm.0005}$	$0.6310_{\pm.0022}$	$0.6395_{\pm.0012}$
w/o agent optimization	$0.0622_{\pm.0006}$	$0.6520_{\pm.0016}$	$0.6502_{\pm.0011}$
w/o joint selection	$0.0650_{\pm.0006}$	$0.6531_{\pm.0020}$	$0.6497_{\pm.0013}$
w/o agent-guided experience	$0.0662_{\pm.0005}$	$0.6575_{\pm.0026}$	$0.6507_{\pm.0012}$

4.2 Ablation Studies

We conduct leave-one-out ablation studies in Table 3 in three domains. Removing agent optimization causes the largest drop on Toys, from 0.0706 to 0.0622. On CAFA, removing harness or specialist optimization reduces NDCG by 0.0438 and 0.0435, respectively. Harness search controls the information exposed to the agent and the experience selected for adaptation, while specialist optimization learns from that experience using task labels. Agent optimization matters most on Toys and specialist optimization on WBM, indicating that their contributions vary by task. Removing agent-guided experience lowers performance even when specialist optimization remains enabled, by 0.0044 on Toys, 0.0170 on CAFA, and 0.0034 on WBM. This decrease supports the role of agent exploration in selecting training experience, in addition to its role in prediction. Removing joint selection also reduces all three scores. Together, we show that all designed components contribute positively to the framework.

4.3 Agent and Model Scaling

Table 4: Scaling experiment on protein function prediction task. NDCG averages cutoffs 5 and 10. Base denotes the initial specialists, which are already trained to convergence before coevolution; Appendix C.3 states the criterion we use. Agent variants use a shared 30-round configuration. Subscripts report standard deviations of within-run gains on three random seeds.

Model	Base	HACE	Reported Δ
Specialist size	Fixed agent Gemma-4 E4B		
ESM-2 35M	0.6283	0.6692	$+0.0409_{\pm0.0007}$
ESM-2 150M	0.6211	0.6633	$+0.0422_{\pm0.0004}$
ESM-2 650M	0.6285	0.6745	$+0.0460_{\pm0.0006}$
ESM-2 3B	0.6339	0.6763	$+0.0424_{\pm0.0007}$
Agent variant	Fixed specialist ESM-2 650M		
Gemma-4 E2B	0.6285	0.6297	$+0.0012_{\pm0.0018}$
Gemma-4 E4B	0.6285	0.6745	$+0.0460_{\pm0.0006}$
Gemma-4 26B-A4B	0.6285	0.6339	$+0.0054_{\pm0.0014}$

In the specialist sweep with the agent fixed, HACE improves prediction with all four ESM-2 sizes in Table 4, from 35M to 3B parameters. The absolute NDCG gains remain between 0.0409 and 0.0460. The largest final score comes from the 3B model, while the largest gain occurs at 650M. The 35M model after HACE also exceeds the initial 3B model, with NDCG of 0.6692 versus 0.6339. In the agent comparison, E4B achieves the largest mean gain of 0.0460, compared with 0.0012 for E2B and 0.0054 for 26B-A4B. These are within-run gains from a separate matched 30-round sweep with ESM-2 650M, detailed in Appendix C.3. The gains do not increase monotonically with agent size, suggesting that a larger agent does not necessarily contribute more to the specialist in this setting.

4.4 Evolution Dynamics

Improvement over rounds. Figure 3 shows HACE reaching the highest score in all three recorded trajectories. The best recorded recommendation score rises mostly in early rounds, while protein and materials reach new maxima after longer intervals. In HACE, each retained specialist and harness defines the starting point for the next round. Fresh agent responses are verified against the cur-

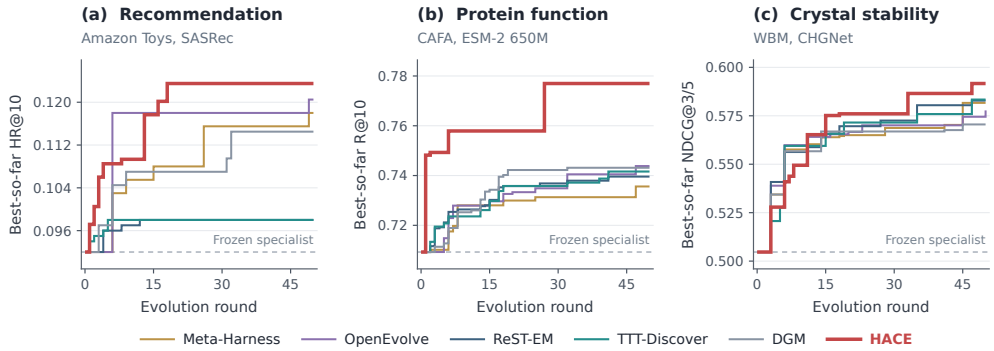

Figure 3: Evolution dynamics of three domains. Best recorded scores during evolution, measured by HR@10, Recall@10, and mean NDCG@3 and NDCG@5. Dashed lines mark the frozen specialist.

rent specialist. This update order makes agent supervision depend on the retained specialist. The trajectories show when higher scoring configurations are found during search.

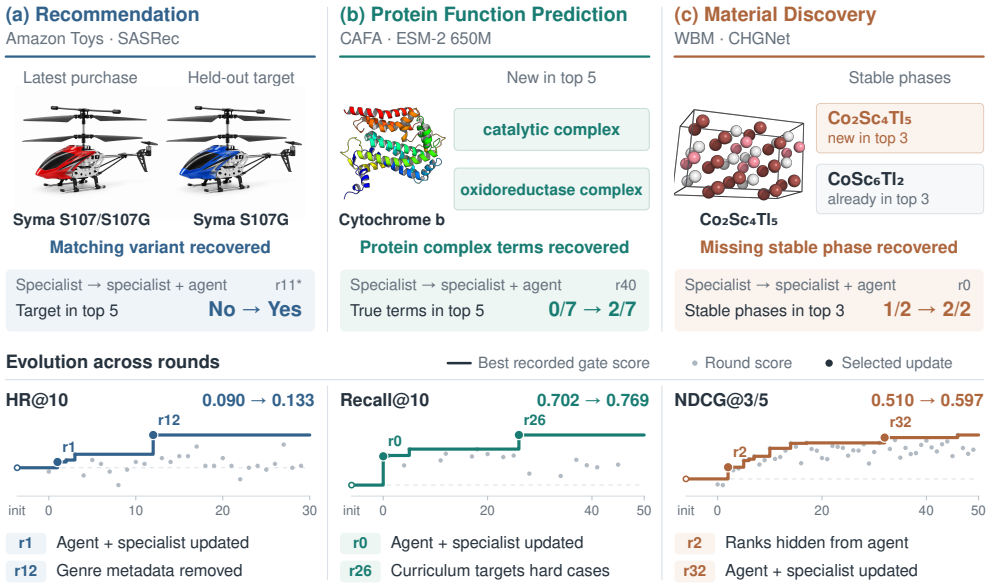

Figure 4: Illustrations on events during coevolution. Upper panels compare specialist and combined predictions. Lower panels show recorded selection scores and their running maxima from one run in each domain, with selected updates annotated.

Case studies on three domains. Figure 4 compares specialist and combined predictions on three domains. HACE recovers a matching product variant, places two relevant protein terms in the top five, and brings a second stable phase into the top three. Recorded changes include removing genre metadata, targeting difficult training instances, and hiding candidate ranks from the agent. These examples illustrate how the harness search and coevolution changes both the context used for prediction and the experience used for reciprocal learning.

5 Conclusion

We introduced HACE, a novel framework for reciprocal self-improvement between specialized foundation models and LLM agents. By evolving both the learners and their interaction with an updating harness, HACE makes each model a source of learning for the other. Our analysis identifies sufficient conditions for specialist learning from adaptively selected experience. Experiments in three domains demonstrate improvements beyond initially converged specialists across different architectures. These results show how heterogeneous AI models can support each other's development. Future work can study sustained coevolution, collaboration among multiple specialists, and autonomous model development for scientific discovery.

AI USE STATEMENT

We used generative AI tools solely to aid and polish the writing of this paper. All AI-assisted edits were reviewed by the authors, who take responsibility for the final content.

ETHICS STATEMENT

Automated model development can inherit biases from training data and evaluation criteria. In a coevolution process, these choices also influence which experiences are selected for further training and which behaviors are reinforced. Improvements on benchmarks do not establish reliability in deployment, and scientific predictions require independent validation before practical use. Applications should consider the quality of the available data and the consequences of incorrect predictions within each domain.

REPRODUCIBILITY STATEMENT

We provide the complete anonymous code for our method in the supplementary materials. The paper and appendices describe the learning objectives, update order, selection procedure, and evaluation metrics that define the coevolution process. These descriptions connect the implementation to the specialist, agent, and harness updates evaluated in our experiments. The code will be open sourced upon acceptance.

REFERENCES

Mert Cemri, Shubham Agrawal, Akshat Gupta, Shu Liu, Audrey Cheng, Qiuyang Mang, Ashwin Naren, Lutfi Eren Erdogan, Koushik Sen, Matei Zaharia, et al. Adaevolve: Adaptive llm driven zeroth-order optimization. *arXiv preprint arXiv:2602.20133*, 2026.

Zhengyu Chen, Teng Xiao, Huaisheng Zhu, Yige Yuan, Luan Zhang, and Jingang Wang. Co-harness: Co-evolving harnesses and model for agents. In *COLM 2026 The 2nd Workshop on Lifelong Agents: Learning, Aligning, and Evolving*, 2026. URL https://openreview.net/forum?id=qDC8LmkpSR.

Bowen Deng, Peichen Zhong, KyuJung Jun, Janosh Riebesell, Kevin Han, Christopher J. Bartel, and Gerbrand Ceder. CHGNet as a pretrained universal neural network potential for charge-informed atomistic modelling. *Nature Machine Intelligence*, 5(9):1031–1041, 2023. doi: 10.1038/s42256-023-00716-3.

Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design. In *Advances in Neural Information Processing Systems*, volume 33, pp. 13049–13061. Curran Associates, Inc., 2020.

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, Debsindhu Bhowmik, and Burkhard Rost. ProtTrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):7112–7127, 2022. doi: 10.1109/TPAMI.2021.3095381.

Gemma Team. Gemma 4 technical report, 2026.

F. Maxwell Harper and Joseph A. Konstan. The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19:1–19:19, 2015. doi: 10.1145/2827872. URL https://doi.org/10.1145/2827872.

Steven R. Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. Time-uniform, nonparametric, nonasymptotic confidence sequences. *The Annals of Statistics*, 49(2):1055–1080, 2021. doi: 10.1214/20-AOS1991.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, 2025.

Daniel H. Janzen. When is it coevolution? *Evolution*, 34(3):611–612, 1980. doi: 10.1111/j.1558-5646.1980.tb04849.x.

Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002. doi: 10.1145/582415.582418.

Yuxiang Jiang, Tal Ronnen Oron, Wyatt T. Clark, et al. An expanded evaluation of protein function prediction methods shows an improvement in accuracy. *Genome Biology*, 17(1):184, 2016. doi: 10.1186/s13059-016-1037-6.

Minki Kang, Jongwon Jeong, Seanie Lee, Jaewoong Cho, and Sung Ju Hwang. Distilling LLM agent into small models with retrieval and code tools, 2025.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206. IEEE, 2018. doi: 10.1109/ICDM.2018.00035.

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-harness: End-to-end optimization of model harnesses. *arXiv preprint arXiv:2603.28052*, 2026.

Zihao Li, Jiaru Zou, Feihao Fang, Xuying Ning, Mengting Ai, Tianxin Wei, Sirui Chen, Xiyuan Yang, and Jingrui He. Heterogeneous scientific foundation model collaboration. *arXiv preprint arXiv:2604.27351*, 2026.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023. doi: 10.1126/science.ade2574.

Jun Liu, Zhenglun Kong, Peiyan Dong, Changdi Yang, Tianqin Li, Yanyue Xie, Yifan Gong, Xuan Shen, Pu Zhao, Hao Tang, Geng Yuan, Wei Niu, Wenbin Zhang, Xue Lin, Yanzhi Wang, and Dong Huang. Structured agent distillation for large language model agents, 2026a.

Shu Liu, Shubham Agarwal, Monishwaran Maheswaran, Mert Cemri, Qiuyang Mang, Zhifei Li, Ashwin Naren, Ethan Boneh, Audrey Cheng, Alexander Du, Melissa Pan, Kurt Keutzer, Alvin Cheung, Koushik Sen, Alex Dimakis, Matei Zaharia, and Ion Stoica. Evox: Meta-evolution for automated discovery. In *Third Conference on Language Modeling*, 2026b.

Ziyang Liu, Xinyan Guo, Xuchen Wei, Han Hao, and Liu Yang. Escher-Loop: Mutual evolution by closed-loop self-referential optimization. arXiv preprint arXiv:2604.23472, 2026c.

Yaxin Luo, Haobin Jiang, Jialv Zou, Xu Huang, Wenhao Yan, Haodong Li, Zhengrong Yue, Jing Li, Xiaofu Chen, Xiaohan Zhao, et al. Autodesign: Meta-harness optimization for long-horizon agentic design. *arXiv preprint arXiv:2608.13560*, 2026.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. In *International Conference on Learning Representations*, 2024.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 43–52, 2015. doi: 10.1145/2766462.2767755. URL https://doi.org/10.1145/2766462.2767755.

Janosh Riebesell, Rhys E. A. Goodall, Philipp Benner, Yuan Chiang, Bowen Deng, Gerbrand Ceder, Mark Asta, Alpha A. Lee, Anubhav Jain, and Kristin A. Persson. A framework to evaluate machine learning crystal stability predictions. *Nature Machine Intelligence*, 7(6):836–847, 2025. doi: 10.1038/s42256-025-01055-1.

Asankhaya Sharma. Openevolve: an open-source evolutionary coding agent, 2025. URL https://github.com/codelion/openevolve.

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.

Hai-Chen Wang, Silvana Botti, and Miguel A. L. Marques. Predicting stable crystalline compounds using chemical similarity. *npj Computational Materials*, 7(1):12, 2021. doi: 10.1038/s41524-020-00481-6. URL https://doi.org/10.1038/s41524-020-00481-6.

Jianze Wang, Ying Liu, Jinlong Chen, Xuchun Hu, Qilong Zhang, Yu Cao, Jun Wang, Hua Yang, Yong Xie, and Qianglong Chen. MAD-OPD: Breaking the ceiling in on-policy distillation via multi-agent debate, 2026.

Rui Wang, Joel Lehman, Jeff Clune, and Kenneth O. Stanley. POET: Open-ended coevolution of environments and their optimized solutions. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 142–151. Association for Computing Machinery, 2019. doi: 10.1145/3321707.3321799.

Shidong Yang, Ziyu Ma, Tongwen Huang, Yiming Hu, Yong Wang, and Xiangxiang Chu. CoEvolve: Training LLM agents via agent-data mutual evolution. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23015–23036. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1055. URL https://aclanthology.org/2026.acl-long.1055/.

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 *International Conference on Machine Learning*, 2026.

Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Jiayuan He, Yinghai Lu, and Yu Shi. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 58484–58509. PMLR, 2024.

Buyun Zhang, Liang Luo, Yuxin Chen, Jade Nie, Xi Liu, Shen Li, Yanli Zhao, Yuchen Hao, Yantao Yao, Ellie Dingqiao Wen, Jongsoo Park, Maxim Naumov, and Wenlin Chen. Wukong: Towards a scaling law for large-scale recommendation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 59421–59434. PMLR, 2024.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin Gödel Machine: Open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, 2026a.

Jenny Zhang, Bingchen Zhao, Wannan Yang, Jakob Foerster, Jeff Clune, Minqi Jiang, Sam Devlin, and Tatiana Shavrina. Hyperagents. arXiv preprint arXiv:2603.19461, 2026b.

Adam Zweiger, Jyo Pari, Han Guo, Yoon Kim, and Pulkit Agrawal. Self-adapting language models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 74084–74115. Curran Associates, Inc., 2025. doi: 10.52202/085713-2483.

APPENDIX

A LIMITATIONS

HACE coevolves a single specialized foundation model with a single LLM agent. We do not study coevolution among multiple specialists or agents, which may require different mechanisms for interaction and learning. The compared methods differ in which components they can update. Some support parameter updates, others evolve the harness or agent program with the specialist held fixed, and several assume a closed-weight LLM. We follow each method's original formulation rather than introducing specialist adaptation beyond its design. These comparisons therefore evaluate the methods as complete systems, while Table 3 separately examines the contribution of each HACE component. Our findings are based on the evaluated domains, architectures, and computational budgets. Further work can study how long coevolution sustains useful learning and how to allocate computation among harness search, specialist adaptation, and agent training.

B PROOFS AND IMPLEMENTATION DETAILS

Throughout this appendix, $\log$ is the natural logarithm and an explicitly written $\log_2$ uses base two. For a finite set or indexed dataset, $|B|$ counts its elements or records, while $|z|$ counts tokens. The indicator $\mathbf{1}\{B\}$ is one when condition B holds and zero otherwise. Empty sums are zero and empty products are one. Unless specified otherwise, $\sum_i$ means $\sum_{i\in\mathcal{I}}$ and $\sum_{t<T}$ means $\sum_{t=0}^{T-1}$.

B.1 SPECIALIST LEARNING

Let $\mathcal{D}_{\text{train}} = \{(x_i, y_i) : i \in \mathcal{I}\}$ be the labeled training pool defined in Section 3.1. Equation 2 gives the supervised objective on the selected training indices $\mathcal{I}_t^S$. The specialist's state includes its trainable parameters ϕ and stored score calibration. Adaptation starts from S_t and uses a fixed optimization budget.

When supervised replay is used, the following term is added to Equation 2, with both terms using original task labels,

$$\mathcal{R}_{\text{replay},t}(\phi) = \sum_{i\in\mathcal{I}} \gamma_{t,i}\ell_{\text{task}}(S_\phi(x_i), y_i). \tag{8}$$

The nonnegative weights $\gamma_{t,i}$ are fixed during the specialist update and are zero unless instance i is retained from earlier specialist training. Their sum sets the total replay strength, and equal positive weights give uniform replay over the retained instances. Setting all weights to zero disables replay. This defines the term without assuming a particular retention policy.

For the proof of Theorem 1, condition on the fixed source pool and initial configuration. The source probabilities μ_i, specialist class $\mathcal{F}$, and losses $\ell_i(f)$ are fixed before exploration. The analysis loss may differ from the optimization loss ℓ_{task}. Interpret $\mathcal{H}_t$ as the sigma field of information available before round t. These histories increase with t, and $\mathcal{H}_{t+1}$ includes that round's proposal, responses, specialist training, validation decision, and agent update.

The accepted weights are measurable with respect to $\mathcal{H}_{t+1}$, are nonnegative, and satisfy $\sum_i W_{t,i} \le 1$. Under the uniform allocation used here, $W_{t,i} = 1/|\mathcal{I}_t^S|$ on selected instances in a nonempty accepted update and zero otherwise. Conditional expectation given $\mathcal{H}_t$ averages over the remaining round randomness, allowing dependence between selection and acceptance. The final specialist may depend on the full observed history. Replay may affect its final loss but is not counted as additional exposure. We use $C_T(i)$, $\widehat{L}_T(f)$, and $U_T(v)$ as defined in Section 3.3.

Proof of Theorem 1. Fix a predictor $f \in \mathcal{F}$ before the random exploration history. Define its accepted loss in round t and the conditional mean of that loss by

$$X_t(f) = \sum_i W_{t,i}\ell_i(f), \qquad m_t(f) = \mathbb{E}[X_t(f) \mid \mathcal{H}_t]. \tag{9}$$

The weight and loss bounds give $0 \le X_t(f) \le 1$ and $0 \le m_t(f) \le 1$. Since the task losses are fixed, linearity of conditional expectation gives

$$\sum_{t<T} m_t(f) = \sum_i C_T(i)\ell_i(f), \qquad \sum_{t<T} X_t(f) = \widehat{L}_T(f). \tag{10}$$

We first compare the conditional expected losses with their realized sum, uniformly over rounds. For $0 \le x \le 1$, convexity gives $e^{-x} \le 1 - (1 - e^{-1})x$. Taking conditional expectations and using $1 - u \le e^{-u}$ yields

$$\mathbb{E}[e^{-X_t(f)} \mid \mathcal{H}_t] \le 1 - (1 - e^{-1})m_t(f) \le e^{-(1-e^{-1})m_t(f)}. \tag{11}$$

Define $M_0(f) = 1$ and

$$M_T(f) = \exp\left((1 - e^{-1}) \sum_{t<T} m_t(f) - \sum_{t<T} X_t(f)\right). \tag{12}$$

This process is nonnegative and measurable with respect to $\mathcal{H}_T$. Equation 11 implies

$$\begin{aligned} \mathbb{E}[M_{T+1}(f) \mid \mathcal{H}_T] &= M_T(f)e^{(1-e^{-1})m_T(f)}\mathbb{E}[e^{-X_T(f)} \mid \mathcal{H}_T] \\ &\le M_T(f). \end{aligned} \tag{13}$$

Thus it is a supermartingale, meaning that its conditional expected next value is at most its current value. This exponential construction gives a probability bound valid over time (Howard et al., 2021). The following stopping argument establishes that bound directly.

For a threshold $a > 1$, let σ_a be the first integer $T \ge 0$ with $M_T(f) \ge a$, and set $\sigma_a = \infty$ if no such round exists. Whether a crossing has occurred by round t is known from $\mathcal{H}_t$, so σ_a is a stopping time. For a positive integer n, write $\sigma_a \wedge n = \min\{\sigma_a, n\}$. The stopped process satisfies

$$M_{\sigma_a \wedge n}(f) = 1 + \sum_{t=0}^{n-1} \mathbf{1}\{\sigma_a > t\}\big(M_{t+1}(f) - M_t(f)\big). \tag{14}$$

Each indicator is known under $\mathcal{H}_t$, so each expected summand is nonpositive by Equation 13. Hence $\mathbb{E}[M_{\sigma_a \wedge n}(f)] \le 1$. The process is at least a whenever a crossing occurs by n and is nonnegative otherwise. Therefore $a \Pr(\sigma_a \le n) \le 1$. Letting $n \to \infty$ gives

$$\Pr\left(\exists\, T \ge 0 \text{ with } M_T(f) \ge a\right) \le \frac{1}{a}. \tag{15}$$

Choose $a = |\mathcal{F}|/\delta$ and take a union bound over the fixed class. With probability at least $1 - \delta$, no predictor's process crosses this threshold at any round. On this event, taking logarithms and applying Equation 10 gives, for all $f \in \mathcal{F}$ and $T \ge 1$,

$$(1 - e^{-1}) \sum_i C_T(i)\ell_i(f) \le \widehat{L}_T(f) + \log(|\mathcal{F}|/\delta). \tag{16}$$

No independence across rounds is used.

This bound controls task risk wherever accepted exposure is sufficiently large. Fix any $v > 0$ on this event. Instances with $C_T(i) < v\mu_i$ contribute at most $U_T(v)$ to the source risk, because their losses are at most one. For each remaining instance, $\mu_i \le C_T(i)/v$. Summing and using Equation 16 gives

$$R_\mu(f) \le U_T(v) + \frac{1}{v} \sum_i C_T(i)\ell_i(f) \le U_T(v) + \frac{\widehat{L}_T(f) + \log(|\mathcal{F}|/\delta)}{(1 - e^{-1})v}. \tag{17}$$

This deterministic implication holds for every $v > 0$ on the same event. Since that event already covers every predictor and round, we may substitute the adaptive final specialist $S_T \in \mathcal{F}$. The result also holds at any finite stopping time chosen from the observed history. Treating only the retrospectively selected S_T as fixed would not justify the probability bound.

Finally, for any positive levels $v_T \to \infty$ such that $U_T(v_T) \to 0$ and $\widehat{L}_T(S_T)/v_T \to 0$, all terms in the upper bound tend to zero. Nonnegativity then gives $R_\mu(S_T) \to 0$. The levels may depend on the observed history because the same event covers every v. This conclusion applies to trajectories satisfying these conditions. For a fixed finite class, a zero risk limit requires $\min_{f \in \mathcal{F}} R_\mu(f) = 0$. □

The exposure term connects the bound to the training allocation. When v is proportional to T, the term from class size decreases as $1/T$, while the other terms depend on exposure and retained loss. To relate exposure to selection, suppose the nonempty training set has at most $B \geq 1$ instances and uses uniform weights. On the joint event that instance i is selected and the proposal is accepted, its weight is at least $1/B$. Thus

$$\mathbb{E}[W_{t,i} \mid \mathcal{H}_t] \geq \frac{1}{B} \Pr\left(i \in \mathcal{I}_t^S,\ \Delta J_t > 0 \mid \mathcal{H}_t\right). \tag{18}$$

Summing gives the corresponding lower bound for $C_T(i)$. The joint probability preserves dependence between inclusion and acceptance. A high overall acceptance rate need not expose every instance, and verification need not lead to inclusion after curriculum truncation.

Small losses immediately after each update do not ensure small final loss when later updates forget those instances. Replay may affect this final loss. If acceptance has conditional probability zero in every round, then $C_T(i) = 0$ and $U_T(v) = 1$, so the bound gives no learning conclusion. Expected exposure is generally not observable from one trace, and $|\mathcal{F}|$ counts a class fixed before exploration rather than adaptively visited checkpoints.

A population bound requires assumptions on how the source pool is drawn. Let P be a distribution of labeled task instances. Suppose the source pool consists of $n = |\mathcal{I}| \geq 1$ independent draws $Z_i = (x_i, y_i)$ from P, indexed by $i = 1, \ldots, n$, and set $\mu_i = 1/n$. Assume $\mathcal{F}$ is fixed independently of these draws. For a generic draw $Z \sim P$, let $\ell(f, Z) \in [0, 1]$ be a fixed loss and set $\ell_i(f) = \ell(f, Z_i)$. Define

$$R_P(f) = \mathbb{E}_{Z \sim P}[\ell(f, Z)], \qquad R_\mu(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f, Z_i). \tag{19}$$

Under the remaining assumptions of Theorem 1, with probability at least $1 - \delta$ over the pool and learning process, every final specialist satisfies, simultaneously for all $T \geq 1$ and $v > 0$,

$$R_P(S_T) \leq U_T(v) + \frac{\widehat{L}_T(S_T) + \log(2|\mathcal{F}|/\delta)}{(1 - e^{-1})v} + \sqrt{\frac{\log(2|\mathcal{F}|/\delta)}{2n}}. \tag{20}$$

Proof. For this calculation, let ξ be a centered random variable supported in an interval of width one and let $\eta \geq 0$ be an exponential moment parameter. Define $\psi(\eta) = \log \mathbb{E}[e^{\eta \xi}]$. Differentiation shows that $\psi''(\eta)$ is the variance of ξ under probabilities reweighted by $e^{\eta \xi}$. This distribution retains the same support, and its variance is at most the expected squared distance to the interval midpoint, hence at most $1/4$. Since $\psi(0) = \psi'(0) = 0$, integrating twice gives

$$\psi(\eta) = \int_0^\eta (\eta - s)\psi''(s)\, ds \leq \eta^2/8, \tag{21}$$

where s is the integration variable. Apply this inequality to $R_P(f) - \ell(f, Z_i)$ for a fixed f. Independence and exponential Markov inequality give, for every $\epsilon > 0$ and $\eta > 0$,

$$\Pr(R_P(f) - R_\mu(f) \geq \epsilon) \leq \exp(-\eta n \epsilon + n\eta^2/8). \tag{22}$$

Choosing $\eta = 4\epsilon$ gives an upper bound of $e^{-2n\epsilon^2}$. Taking $\epsilon = \sqrt{\log(2|\mathcal{F}|/\delta)/(2n)}$ and a union bound over $\mathcal{F}$ therefore yields, with probability at least $1 - \delta/2$,

$$R_P(f) \leq R_\mu(f) + \sqrt{\frac{\log(2|\mathcal{F}|/\delta)}{2n}} \quad \text{for every } f \in \mathcal{F}. \tag{23}$$

Conditional on the realized pool and initial configuration, apply Theorem 1 with failure probability $\delta/2$. Averaging over the pool and initial configuration keeps its failure probability at most $\delta/2$. A union bound combines this event with Equation 23, without requiring independence between the events. On their intersection, which has probability at least $1 - \delta$, adding the bounds proves Equation 20 for every T and v.

Repeated training on a fixed pool does not reduce the final sample term. Population consistency additionally requires that term to vanish, for example by increasing independent source data while controlling class size. The source risk result permits a class chosen after fixing the pool but before exploration. This population extension requires the stronger independence condition stated above.

B.2 Agent Learning and Verification

We next analyze the responses that provide learning experience, conditioning on the collection state and using the fixed source probabilities μ_i. Write A_θ for the agent language model with trainable parameters θ, including an adapter when used, and let θ_t be the parameters of A_t. For a fixed specialist and harness, p_i is the nonempty prompt constructed from x_i and specialist predictions, excluding y_i. The decoded distribution $\pi_\theta(\tau \mid p_i)$ gives the probability of the complete response sequence τ after decoding and stopping. We collect $m \geq 1$ responses per instance, where m is an integer. They are independent conditional on the prompt and any shared task context.

Fix a collection round t, its specialist and harness, and the task context used to form each prompt. For an instance i with evaluable supervision, define

$$E_t(i) = \{\tau \in \mathcal{T} \mid \tau \text{ is valid and } r_i(\tau) > 0\}, \qquad \alpha_t(i) = \pi_{\theta_t}(E_t(i) \mid p_i). \tag{24}$$

Here $\mathcal{T}$ is the countable space of finite response token sequences. For any response distribution π, the notation $\pi(E)$ means $\sum_{\tau \in E} \pi(\tau)$. For an instance without evaluable supervision, set $E_t(i) = \varnothing$ and $\alpha_t(i) = 0$.

Let I be the source instance index, with $\Pr(I = i) = \mu_i$, and let V be the event that at least one of its m collected responses passes verification. Conditional independence of those responses gives $\Pr(V \mid I = i) = 1 - (1 - \alpha_t(i))^m$. Therefore, whenever $\Pr(V) > 0$, Bayes' rule gives

$$\Pr(I = i \mid V) = \frac{\mu_i[1 - (1 - \alpha_t(i))^m]}{\sum_j \mu_j[1 - (1 - \alpha_t(j))^m]}. \tag{25}$$

This describes source sampling followed only by verification. A zero denominator means that no source instance can provide a verified response under this collection procedure.

If the responses share a random context C, independence must instead hold conditional on that context. Define $\alpha_t(i, C)$ by Equation 24 using the prompt and verifier determined by C. The collection probability becomes

$$\Pr(V \mid I = i) = \mathbb{E}\big[1 - (1 - \alpha_t(i, C))^m \mid I = i\big]. \tag{26}$$

Equation 25 then uses this conditional expectation in place of each bracketed collection probability. For $m > 1$, the function $a \mapsto 1 - (1 - a)^m$ is concave on $[0, 1]$, since its second derivative is $-m(m - 1)(1 - a)^{m-2}$ in the interior. Thus Jensen's inequality yields

$$\Pr(V \mid I = i) \leq 1 - \big(1 - \mathbb{E}[\alpha_t(i, C) \mid I = i]\big)^m. \tag{27}$$

Averaging the response probability before computing the collection probability can therefore overestimate the chance of verification. Equality holds for $m = 1$.

We next compare verification before agent fitting, after fitting under the same reference, and after the next specialist and harness selection. Fix one instance and its agent collection context under the retained pair $(\bar{S}_t, \bar{H}_t)$, and suppress the instance index. Write E_t for its verified response set, and let π_t and $\widetilde{\pi}_t$ be the decoded response distributions before and after agent fitting under that same prompt, specialist, and harness. Set $\alpha_t = \pi_t(E_t) > 0$ and $\widetilde{\alpha}_t = \widetilde{\pi}_t(E_t) > 0$.

Let q_t be a distribution on selected responses in E_t, including their training weights. Every response with positive q_t mass must have positive probability under both π_t and $\widetilde{\pi}_t$. For distributions q and p, define $\mathrm{KL}(q\|p) = \sum_{\tau : q(\tau) > 0} q(\tau) \log[q(\tau)/p(\tau)]$, with value $+\infty$ if p is zero on a response with positive q mass. All divergences and log likelihood expectations below are assumed finite. Define

$$g_t = \mathbb{E}_{q_t} \log \frac{\widetilde{\pi}_t(\tau)}{\pi_t(\tau)}, \qquad k_t = \mathrm{KL}\big(q_t \| \pi_t(\cdot \mid E_t)\big), \tag{28}$$

where $\pi_t(\tau \mid E_t) = \pi_t(\tau)/\alpha_t$ on E_t and is zero elsewhere. The conditional distribution after fitting is defined in the same way. Thus g_t measures the mean decoded likelihood gain on selected responses, while k_t measures how their distribution differs from the old distribution conditional on verification.

The next agent collection uses the fitted agent after the next specialist and harness selection. Let π_{t+1} and E_{t+1} be its decoded distribution and verified response set for the same instance, and set

$\alpha_{t+1} = \pi_{t+1}(E_{t+1})$. These probabilities condition on the realized parameters, specialist, harness, and collection context. Define

$$\eta_t = \mathrm{TV}(\widetilde{\pi}_t, \pi_{t+1}) + \widetilde{\pi}_t(E_t \setminus E_{t+1}), \tag{29}$$

where $\mathrm{TV}(p, q) = \sup_{B \subseteq \mathcal{T}} |p(B) - q(B)|$. The two terms measure the change in response distribution and the old probability of responses that no longer pass verification.

Proposition 1 (Agent Verification)**.** *Under the conditions above,*

$$\log \frac{\widetilde{\alpha}_t}{\alpha_t} = g_t - k_t + \mathrm{KL}\big(q_t \| \widetilde{\pi}_t(\cdot \mid E_t)\big) \geq g_t - k_t, \tag{30}$$

and

$$\alpha_{t+1} \geq \big[\alpha_t e^{g_t - k_t} - \eta_t\big]_+, \tag{31}$$

where $[x]_+ = \max\{x, 0\}$*. In particular,* $g_t > k_t$ *is sufficient for improved verification under the fixed reference. The stronger condition* $\eta_t < \alpha_t(e^{g_t - k_t} - 1)$ *is sufficient for* $\alpha_{t+1} > \alpha_t$ *after the reference changes.*

Proof. Expanding the conditional distributions gives

$$k_t = \mathbb{E}_{q_t} \log \frac{q_t(\tau)}{\pi_t(\tau)} + \log \alpha_t,$$
$$\mathrm{KL}\big(q_t \| \widetilde{\pi}_t(\cdot \mid E_t)\big) = \mathbb{E}_{q_t} \log \frac{q_t(\tau)}{\widetilde{\pi}_t(\tau)} + \log \widetilde{\alpha}_t.$$

Subtracting the first equality from the second proves the identity in Equation 30. To prove its inequality, concavity of the logarithm gives, for any distributions q and p with the stated support and integrability conditions,

$$\mathbb{E}_q \log \frac{p(\tau)}{q(\tau)} \leq \log \sum_{\tau : q(\tau) > 0} p(\tau) \leq 0. \tag{32}$$

Hence the KL divergence is nonnegative. Exponentiating yields $\widetilde{\alpha}_t \geq \alpha_t e^{g_t - k_t}$.

For the next collection, total variation and the decomposition of E_t imply

$$\begin{aligned} \alpha_{t+1} &\geq \widetilde{\pi}_t(E_{t+1}) - \mathrm{TV}(\widetilde{\pi}_t, \pi_{t+1}) \\ &\geq \widetilde{\alpha}_t - \widetilde{\pi}_t(E_t \setminus E_{t+1}) - \mathrm{TV}(\widetilde{\pi}_t, \pi_{t+1}) \\ &= \widetilde{\alpha}_t - \eta_t. \end{aligned}$$

Combining this inequality with the fixed reference bound and probability nonnegativity proves Equation 31. The two strict improvement conditions follow directly. □

If $q_t = \pi_t(\cdot \mid E_t)$, then $k_t = 0$. Response selection and length weights generally change this distribution. For m conditionally independent responses in the next collection, monotonicity of $a \mapsto 1 - (1 - a)^m$ also gives

$$\begin{aligned} &\Pr(\text{at least one verified response in round } t+1) \\ &\quad \geq 1 - \Big(1 - \big[\alpha_t e^{g_t - k_t} - \eta_t\big]_+\Big)^m. \end{aligned} \tag{33}$$

The probability conditions on the fixed instance, collection state, and context. The lower bound inside the parentheses lies in $[0, 1]$ because $\eta_t \geq 0$ and $\alpha_t e^{g_t - k_t} \leq \widetilde{\alpha}_t \leq 1$. With random shared context, apply this inequality conditionally and then average as in Equation 26.

To connect this result to the training update, we decompose Equation 5 into its prompt and response terms. The dataset $\mathcal{Z}_t$, token indices, and raw next-token probabilities p_θ are defined in Section 3.2. The loss includes prompt tokens after the first token and all response tokens present in the record, including a stopping token when present. Contribution scores select records without weighting their losses, and the dataset remains fixed during fitting.

Separating prompt and response tokens shows which likelihood gains this objective measures. Fix a nonempty training dataset $\mathcal{Z}$ and suppress the update round. Let θ and $\widetilde{\theta}$ be the parameters before

and after fitting. Its indexed records are $z_j = p_j \| \tau_j$ for $j = 1, \ldots, |\mathcal{Z}|$, where $\|$ denotes token concatenation. Each record has at least two tokens. Let $w_j = 1/(|z_j| - 1)$. The prompt p_j is nonempty, and $p_{j,\ell}$ and $p_{j,<\ell}$ denote its token at position ℓ and preceding prefix. The same indexing applies to z_j. Define the raw response token product

$$P_\theta(\tau_j \mid p_j) = \prod_{\ell=|p_j|+1}^{|z_j|} p_\theta(z_{j,\ell} \mid z_{j,<\ell}). \tag{34}$$

Splitting the predicted tokens at the prompt boundary rewrites Equation 5 exactly as

$$\mathcal{L}_A(\theta) = -\frac{1}{|\mathcal{Z}|} \sum_j w_j \left[\sum_{\ell=2}^{|p_j|} \log p_\theta(p_{j,\ell} \mid p_{j,<\ell}) + \log P_\theta(\tau_j \mid p_j) \right]. \tag{35}$$

An empty prompt token sum is zero. The first token has no preceding token in the training sequence and is excluded from the implemented loss. Assume that both parameter values assign positive probabilities to every predicted token in these records. The dataset and its length weights remain fixed during this comparison.

Partition the record indices into groups $\mathcal{J}_u$ with the same source instance, prompt, and verifier, where u ranges over the nonempty groups. Write p_u for their common prompt. Their weighted response distributions and shares of total weight are

$$q_u(\tau) = \frac{\sum_{j \in \mathcal{J}_u,\, \tau_j = \tau} w_j}{\sum_{j \in \mathcal{J}_u} w_j}, \qquad \rho_u = \frac{\sum_{j \in \mathcal{J}_u} w_j}{\sum_j w_j}. \tag{36}$$

Thus $\sum_u \rho_u = 1$, and each q_u has finite support consisting of responses that pass one fixed verifier. Subtracting the two losses and collecting response terms by group gives

$$\begin{aligned} \mathcal{L}_A(\theta) - \mathcal{L}_A(\widetilde{\theta}) = {} & \frac{1}{|\mathcal{Z}|} \sum_j w_j \sum_{\ell=2}^{|p_j|} \log \frac{p_{\widetilde{\theta}}(p_{j,\ell} \mid p_{j,<\ell})}{p_\theta(p_{j,\ell} \mid p_{j,<\ell})} \\ & + \frac{\sum_j w_j}{|\mathcal{Z}|} \sum_u \rho_u \mathbb{E}_{q_u} \log \frac{P_{\widetilde{\theta}}(\tau \mid p_u)}{P_\theta(\tau \mid p_u)}. \end{aligned} \tag{37}$$

The first term accounts for fitting prompt tokens. The second averages response gains over groups, with larger record weights for shorter concatenations. A lower total loss therefore does not establish an improved response likelihood for each instance.

The proposition uses decoded probabilities π_θ, whereas the loss uses raw token probabilities p_θ. If both decoded distributions are positive on the finite support of q_u, direct cancellation gives

$$\begin{aligned} \mathbb{E}_{q_u} \log \frac{\pi_{\widetilde{\theta}}(\tau \mid p_u)}{\pi_\theta(\tau \mid p_u)} = {} & \mathbb{E}_{q_u} \log \frac{P_{\widetilde{\theta}}(\tau \mid p_u)}{P_\theta(\tau \mid p_u)} \\ & + \mathbb{E}_{q_u} \log \frac{\pi_{\widetilde{\theta}}(\tau \mid p_u) P_\theta(\tau \mid p_u)}{\pi_\theta(\tau \mid p_u) P_{\widetilde{\theta}}(\tau \mid p_u)}. \end{aligned} \tag{38}$$

The left side is the decoded likelihood gain g_t for that group. The final term is zero when decoding samples directly from the model and the recorded token product contains every generation probability factor, including any stopping token. Temperature scaling, probability truncation, or an omitted stopping factor can change it. If decoding gives zero probability to a selected response, the finite likelihood and divergence conditions do not hold.

Proposition 1 explains how agent fitting and later reference changes affect verification. The curriculum then selects specialist training instances, and pair acceptance determines their contribution to the exposure weights in Theorem 1. Increased verification alone does not establish that theorem's exposure or retained loss conditions.

B.3 Task and Implementation Details

Our experiments use candidate predictions in recommendation, protein function prediction, and material discovery. For an instance i, let $\mathcal{C}_i$ be the fixed reference candidate pool and $Y_i \subseteq \mathcal{C}_i$ the

relevant candidates determined by the original label y_i. The harness exposes a view $\mathcal{C}_i^{S,H} \subseteq \mathcal{C}_i$. The definitions below apply to training and validation instances alike.

To evaluate an individual response, the parser extracts an ordered list of distinct candidate indices from the view. A response is valid if this list is nonempty. It is then completed with omitted candidates in specialist order. A response with no valid indices receives quality zero without completion. For any ordering ρ of distinct candidates, let $|\rho|$ be its length and $\rho(r)$ the candidate at rank r. At a positive integer cutoff k, binary normalized discounted cumulative gain is (Järvelin & Kekäläinen, 2002)

$$\mathrm{NDCG}_k(\rho, Y_i) = \begin{cases} \dfrac{\sum_{r=1}^{\min(k,|\rho|)} \mathbf{1}\{\rho(r) \in Y_i\} / \log_2(r+1)}{\sum_{r=1}^{\min(k,|Y_i|)} 1/\log_2(r+1)}, & Y_i \neq \varnothing, \\ 0, & Y_i = \varnothing. \end{cases} \tag{39}$$

For a valid response τ, its quality $q_i^A(\tau)$ is this score on its completed ordering at the harness's verification cutoff k_H. The specialist quality q_i^S is the same score on the specialist ordering of the same view. Verification requires $Y_i \neq \varnothing$ and $q_i^A(\tau) > q_i^S$. An empty view has no valid responses. Empty target sets are excluded from verification and calibration fitting, but receive zero in system evaluation.

The normalization uses all relevant candidates in the reference pool, so removing relevant candidates from a view does not lower the denominator. When $Y_i \neq \varnothing$,

$$q_i^A(\tau) \leq \frac{\sum_{r=1}^{\min(k_H, |Y_i \cap \mathcal{C}_i^{S,H}|)} 1/\log_2(r+1)}{\sum_{r=1}^{\min(k_H,|Y_i|)} 1/\log_2(r+1)}. \tag{40}$$

At most $|Y_i \cap \mathcal{C}_i^{S,H}|$ relevant candidates are available, and placing them first maximizes the discounted gain. This proves the bound and shows it is attainable by an ordering of the view. If the specialist attains this ceiling, further response sampling in that view cannot produce a positive contribution. Changing the harness can change the available view.

Joint selection orders each view by the combined score in Equation 3. The agent ranking is the ordered candidate list returned by the evaluation response procedure, and specialist scores include any stored calibration. Each compared state fits λ separately over the grid $\{0, 0.25, 0.5, 1, 2, 4, 8\}$. Let $\mathcal{K}$ be the task's finite nonempty set of positive integer evaluation cutoffs. Let $\mathcal{D}_{\mathrm{val}}$ and $\mathcal{D}_{\mathrm{test}}$ be the validation and test pools, both disjoint from $\mathcal{D}_{\mathrm{train}}$ and from each other. The comparison indices $\mathcal{B}_t$ refer to a batch from $\mathcal{D}_{\mathrm{val}}$. Split this batch into the same two nonempty folds for both compared states. On each fitting fold, choose the grid value maximizing mean NDCG averaged over $\mathcal{K}$, using only instances with nonempty target sets. Use zero if the fitting fold has no such instances. Evaluate on the other fold, then exchange their roles. All parsing, ordering, and tie rules remain fixed during the comparison. At zero weight, the joint ordering agrees with the specialist ordering within the same view.

Let ρ_i be the joint ordering for evaluation instance i, formed with the weight fitted on the opposite fold. Then the score in Equation 4 is

$$J_{\mathcal{B}_t}(S, H, A) = \frac{1}{|\mathcal{B}_t|} \sum_{i \in \mathcal{B}_t} \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \mathrm{NDCG}_k(\rho_i, Y_i). \tag{41}$$

The orderings depend on the evaluated (S, H, A) and that state's fitted weights. This instance average weights the fold means by fold size and includes empty target sets with their zero scores. Pair selection compares joint predictions, while response verification compares an individual interpreted response with the specialist.

These evaluation rules determine the round procedure. Sample two archived harnesses and select the one with the higher score recorded at acceptance. Use a single entry directly, or the current harness if the archive is empty. Mutate one or two categorical fields to obtain H'. Under (S_t, H', A_t), the verified curriculum prioritizes instances with at least one verified response, and the unsuccessful curriculum prioritizes those with none. Both select in pool order, truncate at the training budget, and fill shortfalls from remaining instances. The uncertainty curriculum prefers smaller differences between the two largest scores $s_i(c) + a_i(c)$. Uniform sampling bypasses exploration. The selected instances train S' with Equation 2.

With A_t fixed, evaluate the proposed and current pairs using Equation 41. Any specialist calibration uses a validation draw disjoint from this comparison batch. Accept a strictly positive improvement and record the harness and score, or restore the specialist parameters, optimizer, calibration, and harness.

The retained pair then supplies the context for agent learning. Under $(\bar{S}_t, \bar{H}_t)$, collect fresh responses with A_t. Keep each eligible instance's original response with the largest positive contribution. The agent curriculum retains all such pairs or the half with the largest contributions, keeping at least one when available. The retained original tokens form $\mathcal{Z}_t$ for Equation 5. An empty dataset leaves the agent unchanged. Set $S_{t+1} = \bar{S}_t$ and $H_{t+1} = \bar{H}_t$. The next round starts from $(S_{t+1}, H_{t+1}, A_{t+1})$. Labels enter specialist supervision, verification, and evaluation, and are excluded from prompts. Validation data are reused across rounds, and test data are reserved for final evaluation.

Figure 2 uses $\mathcal{D}_S$ for the labeled examples indexed by $\mathcal{I}_t^S$. An archive entry (H_i, J_i) stores a harness and its score at acceptance, where i indexes archive entries in that panel. The figure abbreviates $J_{\mathcal{B}_t}(S, H, A_t)$ as $J(S, H; A_t)$ and ΔJ_t as ΔJ. Sampling $\tau \sim A_t$ means sampling from $\pi_{\theta_t}(\cdot \mid p_i)$ under the retained pair. Superscripts in $S^{(k)}$, $H^{(k)}$, and $A^{(k)}$ count component versions. The specialist and harness advance on acceptance, and the agent advances when it is updated.

C Additional Experiments

C.1 Evaluation Details

We fix disjoint training, validation, and test sets before coevolution. Specialist adaptation, agent learning, and training instance selection use only $\mathcal{D}_{\text{train}}$. At each round, we compare the proposed and current specialist and harness pairs on the same batch sampled from $\mathcal{D}_{\text{val}}$, holding the agent fixed. Both pairs use the same two folds, but fit their own λ on one fold and are evaluated on the other. We then exchange the folds and average the resulting instance scores for joint selection. Internal specialist calibration uses a separate validation batch disjoint from the comparison batch. Validation data guide selection throughout coevolution, while $\mathcal{D}_{\text{test}}$ is reserved for final evaluation and does not inform training, calibration, or selection.

For recommendation, we partition users across the three sets and train the initial specialist on histories excluding each user's last two interactions. During coevolution, we use the second to last interactions of training users for learning and those of validation users for calibration and selection. Final evaluation uses the last interaction of each test user. For protein function prediction, we follow the CAFA temporal split, restrict inputs to information available at t_0, and keep instances sharing a protein name in the same set. For material discovery, we partition by chemical system, keeping all candidate structures from the same system in one set. Below we describe the evaluation metrics.

Ranking metrics. Hit rate at cutoff k (HR@k) is the fraction of recommendation queries whose held out target item appears among the first k predictions. We report HR@10 in the main comparison. Normalized discounted cumulative gain (NDCG) assigns a binary relevance label to each candidate. For a ranking π, its discounted cumulative gain is

$$\text{DCG@}k(\pi) = \sum_{r=1}^{\min(k,|\pi|)} \frac{\text{rel}(\pi_r)}{\log_2(r+1)}. \tag{42}$$

NDCG@k divides this sum by the largest possible DCG@k within the same fixed reference pool. It rewards both recovering relevant candidates and placing them earlier. The reported CAFA score averages NDCG@5 and NDCG@10, while the WBM score averages NDCG@3 and NDCG@5.

Protein function metrics. Precision is the fraction of predicted function terms that are correct, and recall is the fraction of relevant terms recovered. Recall@10 measures the fraction of relevant terms recovered among the first ten predictions. At prediction threshold t, their harmonic mean is $F_1(t) = 2P(t)R(t)/(P(t) + R(t))$. The maximum score over thresholds is $F_{\max} = \max_t F_1(t)$. The area under the precision–recall curve (AUPR) summarizes the tradeoff between precision and recall across thresholds. Minimum semantic distance ($S_{\min}$) measures errors using the information

added by each ontology term given its parents (Jiang et al., 2016). At each threshold, remaining uncertainty $\mathrm{ru}(t)$ is the average information in missed terms, and misinformation $\mathrm{mi}(t)$ is the average information in incorrectly predicted terms. The metric is

$$S_{\min} = \min_t \sqrt{\mathrm{ru}(t)^2 + \mathrm{mi}(t)^2}. \tag{43}$$

Lower values indicate less semantic error.

Materials metrics. For crystal stability prediction, F_1 is the harmonic mean of precision and recall for the stable class. The discovery acceleration factor (DAF) divides precision by the fraction of stable structures in the evaluation pool (Riebesell et al., 2025). A value of one matches the expected precision of random selection. Mean absolute error (MAE) averages the absolute differences between predicted and reference target values. The coefficient of determination (R^2) equals one minus the ratio of squared prediction error to the squared error of predicting the mean reference value. It equals one for perfect predictions, zero for this constant baseline, and is negative when predictions have greater squared error. Higher values are better for all metrics except $S_{\min}$ and MAE.

C.2 Complete Scientific Results

Table 5: Protein function prediction on CAFA with two frozen backbones.

Method	$F_{\max}\uparrow$	AUPR↑	$S_{\min}\downarrow$	NDCG↑	Recall@10↑
ESM-2					
Base	$0.2218_{\pm 0.0014}$	$0.1078_{\pm 0.0012}$	$12.557_{\pm 0.060}$	$0.6285_{\pm 0.0019}$	$0.7179_{\pm 0.0014}$
ReST-EM	$0.2219_{\pm 0.0014}$	$0.1079_{\pm 0.0012}$	$12.557_{\pm 0.060}$	$0.6286_{\pm 0.0018}$	$0.7180_{\pm 0.0012}$
Meta-Harness	$0.2224_{\pm 0.0013}$	$0.1076_{\pm 0.0012}$	$12.553_{\pm 0.059}$	$0.6309_{\pm 0.0015}$	$0.7245_{\pm 0.0007}$
OpenEvolve	$0.2230_{\pm 0.0012}$	$0.1087_{\pm 0.0011}$	$12.549_{\pm 0.060}$	$0.6320_{\pm 0.0018}$	$0.7222_{\pm 0.0016}$
DGM	$0.2219_{\pm 0.0014}$	$0.1075_{\pm 0.0012}$	$12.557_{\pm 0.060}$	$0.6325_{\pm 0.0019}$	$0.7261_{\pm 0.0016}$
TTT-Discover	$0.2219_{\pm 0.0014}$	$0.1078_{\pm 0.0013}$	$12.557_{\pm 0.060}$	$0.6287_{\pm 0.0018}$	$0.7185_{\pm 0.0013}$
HACE	$\mathbf{0.2438}_{\pm 0.0019}$	$\mathbf{0.1258}_{\pm 0.0015}$	$\mathbf{12.494}_{\pm 0.048}$	$\mathbf{0.6745}_{\pm 0.0016}$	$\mathbf{0.7615}_{\pm 0.0015}$
ProtT5-XL					
Base	$0.2221_{\pm 0.0005}$	$0.1078_{\pm 0.0002}$	$13.127_{\pm 0.074}$	$0.6241_{\pm 0.0011}$	$0.7122_{\pm 0.0015}$
ReST-EM	$0.2221_{\pm 0.0005}$	$0.1079_{\pm 0.0003}$	$13.127_{\pm 0.074}$	$0.6242_{\pm 0.0011}$	$0.7131_{\pm 0.0016}$
Meta-Harness	$0.2229_{\pm 0.0006}$	$0.1085_{\pm 0.0002}$	$13.122_{\pm 0.075}$	$0.6261_{\pm 0.0017}$	$0.7156_{\pm 0.0020}$
OpenEvolve	$0.2230_{\pm 0.0003}$	$0.1082_{\pm 0.0003}$	$13.121_{\pm 0.072}$	$0.6254_{\pm 0.0014}$	$0.7139_{\pm 0.0015}$
DGM	$0.2235_{\pm 0.0006}$	$0.1088_{\pm 0.0001}$	$\mathbf{13.118}_{\pm 0.075}$	$0.6273_{\pm 0.0016}$	$0.7175_{\pm 0.0016}$
TTT-Discover	$0.2221_{\pm 0.0005}$	$0.1079_{\pm 0.0003}$	$13.127_{\pm 0.074}$	$0.6241_{\pm 0.0011}$	$0.7135_{\pm 0.0016}$
HACE	$\mathbf{0.2317}_{\pm 0.0005}$	$\mathbf{0.1177}_{\pm 0.0010}$	$13.127_{\pm 0.072}$	$\mathbf{0.6739}_{\pm 0.0013}$	$\mathbf{0.7553}_{\pm 0.0012}$

Table 6: Material discovery task on WBM with CHGNet.

Method	NDCG↑	$F_1\uparrow$	DAF↑	MAE↓	$R^2\uparrow$
Base	$0.6388_{\pm 0.0011}$	$0.5159_{\pm 0.0003}$	$1.644_{\pm 0.003}$	$0.1049_{\pm 0.0002}$	$-1.289_{\pm 0.001}$
ReST-EM	$0.6496_{\pm 0.0006}$	$0.5508_{\pm 0.0004}$	$2.172_{\pm 0.005}$	$0.0756_{\pm 0.0002}$	$-0.323_{\pm 0.001}$
Meta-Harness	$0.6513_{\pm 0.0008}$	$0.5510_{\pm 0.0004}$	$2.171_{\pm 0.003}$	$0.0754_{\pm 0.0003}$	$-0.281_{\pm 0.001}$
OpenEvolve	$0.6486_{\pm 0.0008}$	$0.5463_{\pm 0.0003}$	$2.193_{\pm 0.004}$	$0.0755_{\pm 0.0002}$	$-0.264_{\pm 0.001}$
DGM	$0.6496_{\pm 0.0007}$	$0.5508_{\pm 0.0003}$	$2.183_{\pm 0.003}$	$0.0748_{\pm 0.0003}$	$-0.251_{\pm 0.001}$
TTT-Discover	$0.6474_{\pm 0.0007}$	$0.5504_{\pm 0.0003}$	$2.203_{\pm 0.003}$	$0.0748_{\pm 0.0002}$	$-0.284_{\pm 0.001}$
HACE	$\mathbf{0.6541}_{\pm 0.0010}$	$\mathbf{0.5513}_{\pm 0.0004}$	$\mathbf{2.242}_{\pm 0.004}$	$\mathbf{0.0729}_{\pm 0.0003}$	$\mathbf{-0.203}_{\pm 0.001}$

C.3 More Experimental Details

Agent and model scaling. Table 4 varies specialist size and agent size in separate sweeps. ESM-2 backbone representations remain frozen while the prediction head is trained. The agent variants share a 30-round configuration with ESM-2 650M. In this sweep, Base is the initial specialist score, shared across the agent variants, and Δ is the reported mean within-run gain. The specialist sweep

uses a separate configuration. Subscripts report the standard deviation of the gains. The HACE means for ESM-2 35M, 150M, and 3B are obtained by adding the reported Base means and gains. Both the specialist sweep and all agent variants use three random seeds.

Convergence on base. We begin with specialists already trained to convergence under their original schedules. Continued supervised training alone does not by itself indicate which further updates help, and our ablations show that the acceptance gate in Equation 3 is what makes continued training productive. This setting tests whether interaction with an evolving agent can create additional opportunities beyond that plateau. We define convergence as follows: continuing the original supervised schedule for an additional 3 epochs changes the reported metric by less than 0.001 in every domain.

Evolution records. The lower panels of Figure 4 show selection scores from one recorded run in each domain. Grey points are the available per-round scores, and the lines track their running maxima. The recommendation trajectory covers 30 rounds, while the protein and materials trajectories cover 50 rounds. The CAFA record contains scores for 18 rounds. Missing entries are omitted. These trajectories describe the scores observed during selection, while the main tables report aggregate evaluation results. The recommendation case was collected in a separate logging run with the same configuration, seed, and data as its trajectory. The upper panels compare specialist and combined predictions on the same instance. The protein example is recorded at round 40 and the materials example at round 0.

Hyperparameter. We sample 192 training instances per round. For each instance, the agent generates m responses with temperature 1.0 and top-p 0.95, where $m = 8$ for recommendation and crystal stability and $m = 4$ for protein function. The generation batch size is 96. Prompts exceeding 2560 tokens are truncated by retaining the first 853 and the last 1707 tokens. For joint selection, we sample $|\mathcal{B}_t| = \min(256, |\mathcal{D}_{\text{val}}|)$ validation instances. We train the agent with LoRA adapters ($r = 16$, $\alpha = 32$) using AdamW with a learning rate of 10^{-5}, gradient clipping at 1.0, and a micro-batch size of 2 sequences. Specialist adaptation in Equation 2 uses 200 steps for recommendation and crystal stability and 750 steps for protein function, all at a learning rate of 10^{-4}.

Hardware. All experiments are conducted on NVIDIA B200 GPU clusters, where each node contains 8 GPUs.

Table 7: HACE is agnostic to specialist architectures and adaptation on different learning components.

Domain	Model	Architecture	Updated parameters
Recommendation	SASRec	Causal self-attention Transformer	Model parameters
	HSTU	Causal sequential transducer with pointwise aggregated attention	Model parameters
	Wukong	Stacked factorization-machine network	Model parameters
Protein	ESM-2	Encoder-only protein Transformer	Prediction head
	ProtT5-XL	Protein T5 with encoder-decoder pretraining	Prediction head
Materials	CHGNet	Charge-informed graph neural potential	Model parameters

Table 8: How exploration guides the two learners. Relative improvement determines eligibility for different learning objectives.

Exploration outcome	Agent learning	Specialist learning
At least one sampled ranking improves the specialist	Best positive response is eligible for the agent curriculum	Instance is eligible for the verified curriculum
No sampled ranking improves the specialist	No response from this instance enters the agent update	Instance is eligible for the unsuccessful curriculum

Specialist selection uses exploration before joint selection. Agent learning uses fresh responses afterward. The specialist always trains on task labels. Uniform and uncertainty curricula, as well as budget completion, may select instances from either row.

D HARNESS ENGINEERING

D.1 PROMPTS FOR RECOMMENDATION

We use a common recommendation interface for MovieLens 1M and the Amazon Beauty, CDs, Sports, and Toys datasets. For each user, it combines an interaction history with a candidate ranking from SASRec, HSTU, or Wukong. Harness search controls the history length, available item metadata, and the partition between leading predictions and candidate alternatives. The prompt uses the same item terminology across all five datasets.

Prompt: Recommendation Systems

Use the user's interaction history and the available item metadata to select the {k} candidate alternatives that best match the user's preferences. The specialist's leading predictions provide additional context.

Interaction history

{recent interactions in sequence, with item titles and available metadata}

Specialist's leading predictions

{leading items in specialist order, with available metadata}

Candidate alternatives

1. {item title and available metadata}
2. {item title and available metadata}
...

Output

Return exactly {k} distinct IDs from Candidate alternatives, ordered from best to worst and separated by spaces. Return no other text.

When $k = 1$ and the leading block is nonempty, the following instruction uses the same inputs and allows the agent to retain the specialist's selection. Otherwise, the ranking prompt above applies.

Prompt: Recommendation with a Single Selection

Use the user's interaction history and the available item metadata to identify the best candidate alternative. Compare it with the lowest ranked item among the specialist's leading predictions.

Output

If this alternative is a better match for the user, return its integer ID. Otherwise, return 0. Return one integer and no other text.

Table 9: Recommendation harness fields. "Fixed" denotes a value outside the search space.

Field	Initial	Candidate values	Role
`n_candidates`	50	Fixed	Specialist shortlist size
`n_history`	15	5, 10, 15, 25, 40	Number of history entries
`top_k`	5	1, 3, 5, 10	Requested ranking length
`promote_from`	10	0, 5, 10, 20	Leading candidates shown separately
`n_promote`	1	Fixed	Promotion count when $k = 1$
`show_year`	True	True, False	Release year metadata
`show_genre`	True	True, False	Category metadata

D.2 PROMPTS FOR PROTEIN FUNCTION PREDICTION

For protein function prediction, the candidates are Gene Ontology terms. The prompt combines protein metadata and available annotations with the specialist's ranking. Harness search controls the number of annotations, the length of term definitions, and the metadata shown to the agent. The agent returns candidate IDs using the same ranking format as in recommendation.

Prompt: Protein Function Prediction

Use the protein information, available annotations, and Gene Ontology term descriptions to select the {k} candidate alternatives most likely to apply to this protein. The specialist's leading predictions provide additional context.

Protein information

Protein: {protein name}
Organism: {organism, when enabled}
Target ontology aspect: {aspect, when enabled}
Available annotations: {known terms}

Specialist's leading predictions

{GO terms in specialist order, with available definitions}

Candidate alternatives

1. {GO term name and available definition}
2. {GO term name and available definition}
...

Output

Return exactly {k} distinct IDs from Candidate alternatives, ordered from most to least likely and separated by spaces. Return no other text.

Table 10: Presentation fields of the protein harness. Experience selection is described in Appendix B.3.

Field	Initial	Candidate values	Role
`n_candidates`	50	Fixed	Specialist shortlist size
`n_known`	10	0, 5, 10, 20	Number of known annotations
`top_k`	5	1, 3, 5, 10	Requested ranking length
`promote_from`	10	0, 5, 10, 20	Leading candidates shown separately
`def_chars`	120	0, 120, 300	Definition length, zero for name only
`show_aspect`	True	True, False	Target ontology aspect
`show_organism`	True	True, False	Source organism

D.3 Prompts for Material Discovery

For material discovery, the agent compares candidate crystals within a chemical system. The prompt provides compositions and available structure prototypes, together with the specialist's leading predictions. Harness search controls the prototype length and whether element information and specialist ranks are shown. Candidate IDs identify the agent's selections independently of the specialist ranks.

Prompt: Crystal Stability Prediction

Use the chemical system, candidate compositions, and available structural descriptors to select the {k} candidate alternatives most likely to be thermodynamically stable. The specialist's leading predictions provide additional context.

Chemical system

System: {chemical system}
Elements: {element set, when enabled}

Specialist's leading predictions

{formulas in specialist order, with available structure prototypes}

Candidate alternatives

1. {formula, structure prototype, and specialist rank when enabled}
2. {formula, structure prototype, and specialist rank when enabled}
...

Output

Return exactly {k} distinct IDs from Candidate alternatives, ordered from most to least likely and separated by spaces. Use the IDs at the start of the candidate rows. Return no other text.

Table 11: Presentation fields of the crystal stability harness. Prototype lengths are measured in characters.

Field	Initial	Candidate values	Role
`n_candidates`	20	Fixed	Specialist shortlist size
`top_k`	3	1, 2, 3, 5	Requested ranking length
`promote_from`	2	0, 1, 2, 3	Leading candidates shown separately
`proto_chars`	24	0, 24, 48	Structure prototype length
`show_elements`	True	True, False	Element set
`show_rank`	True	True, False	Specialist rank per candidate