
MindForge: Empowering Embodied Agents with Theory of Mind for Lifelong Cultural Learning

Mircea Lică*

Delft University of Technology
mlica@tudelft.nl

Ojas Shirekar*

Delft University of Technology
o.k.shirekar@tudelft.nl

Baptiste Colle

Delft University of Technology
collebaptiste@gmail.com

Chirag Raman

Delft University of Technology
c.a.raman@tudelft.nl

Abstract

Embodied agents powered by large language models (LLMs), such as Voyager, promise open-ended competence in worlds such as Minecraft. However, when powered by open-weight LLMs they still falter on elementary tasks after domain-specific fine-tuning. We propose MINDFORGE, a generative-agent framework for cultural lifelong learning through explicit perspective taking. We introduce three key innovations: (1) a structured theory of mind representation linking percepts, beliefs, desires, and actions; (2) natural inter-agent communication; and (3) a multi-component memory system. Following the cultural learning framework, we test MINDFORGE in both instructive and collaborative settings within Minecraft. In an *instructive* setting with GPT-4, MINDFORGE agents powered by open-weight LLMs significantly outperform their Voyager counterparts in basic tasks yielding $3\times$ more tech-tree milestones and collecting $2.3\times$ more unique items than the Voyager baseline. Furthermore, in fully *collaborative* settings, we find that the performance of two underachieving agents improves with more communication rounds, echoing the Condorcet Jury Theorem. MINDFORGE agents demonstrate sophisticated behaviors, including expert-novice knowledge transfer, collaborative problem solving, and adaptation to out-of-distribution tasks through accumulated cultural experiences.

1 Introduction

The development of generally capable agents marks a significant shift in advancing artificial intelligence, transitioning from assimilating data to generating novel knowledge through embodied interactions with open-ended environments [1–4]. Classical approaches leveraging reinforcement learning [5, 6] and imitation learning [7] often struggle with generalization and exploration, as agents tend to converge on repetitive behaviors in static environments [8]. To address these limitations, researchers have sought to emulate human-like lifelong learning capabilities, developing systems that can continuously acquire, update, and transfer knowledge over extended periods [9, 10]. The advent of large language models (LLMs) has accelerated this pursuit, enabling the development of agents such as Voyager [11] that can apply internet-scale knowledge to continuously explore, plan, and acquire new skills in partially observable, open-ended environments such as Minecraft.

Despite their promise, we argue that state-of-the-art lifelong learning agents like Voyager face a crucial limitation: they learn in isolation, neglecting a fundamental aspect of human intelligence—the

*Equal Contribution.

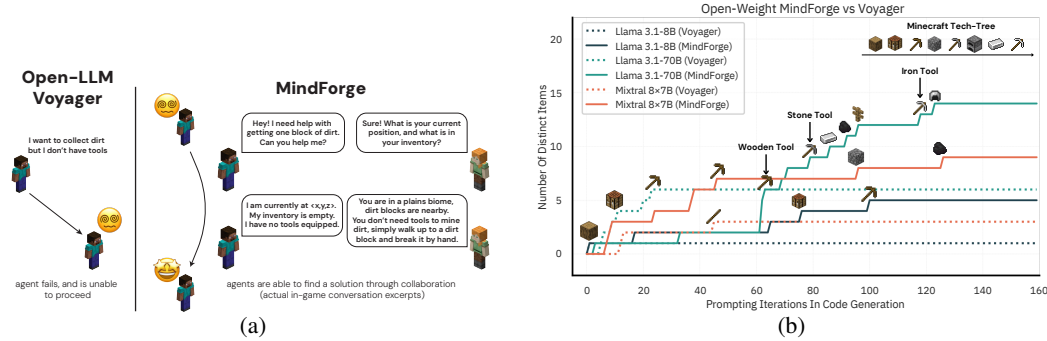


Figure 1: MINDFORGE enables agents to (a) collaborate through structured theory-of-mind representations, which (b) leads to improved performance compared to Voyager when powered by open-weight LLMs, averaged across 3 runs.

social context. So central is the social context to our existence that the Social Intelligence Hypothesis posits that our cognitive capabilities evolved primarily to navigate the complexities of social life [12, 13]. This isolated learning becomes particularly problematic when coupled with these agents’ reliance on closed LLMs like GPT-4. Wang et al. [11] note that “VOYAGER requires the quantum leap in code generation quality from GPT-4 which GPT-3.5 and open-source LLMs cannot provide”. Our experiments (Section 5) corroborate this observation, demonstrating that replacing GPT-4 with open LLMs renders the Voyager agent incapable of even the most rudimentary tasks, such as collecting dirt in Minecraft. We ask: can an open LLM-powered agent, which struggles with basic tasks, improve through collaboration with others?

To address this question, we introduce the MINDFORGE framework, inspired by cultural learning, which contends that individuals internalize skills and strategies by observing others’ perspectives, using Theory of Mind (ToM) to infer goals, beliefs, and intentions and thereby deepen social learning [14–18]. Our framework reframes *knowledge distillation* [19] as a social, test-time process: agents refine their beliefs through multi-turn dialogue with expert peers, without gradient updates. This interactive distillation supports *test-time compute scaling* [20], enabling agents to improve performance via communication, memory retrieval and perspective-taking. By structuring communication across turns, MINDFORGE also contributes to *multi-turn LLM agent* research [21, 22], showing that open models can match GPT-4 through collaboration alone. Moreover, our analysis of weak-weak settings highlights a compute-performance tradeoff at inference, linking collaboration to test-time compute considerations [20].

Figure 1a provides an overview of our agents in action. Concretely, MINDFORGE extends Voyager by incorporating three key improvements: (1) **perspective-taking** through a causal ToM template [23] linking agent percepts, beliefs, desires, and actions; (2) a **natural language communication interface** between agents; and (3) memory updates following the Soar cognitive architecture [24], including **semantic memory** of task and environment-related knowledge and **episodic memory**.

Empirically, we instantiate MINDFORGE agents in *Instructed Learning* (mixed-expertise) and *Collaborative Learning* (matched-expertise) settings within the cultural learning framework [15]. MINDFORGE agents powered by open-weight LLMs outperform the original Voyager architecture [11] on two seemingly trivial tasks by more than 40%. We identify Voyager’s specific failure cases: (1) **false beliefs** and (2) **code generation**, and show how our method overcomes them through communication and perspective-taking. To assess lifelong learning, we run MINDFORGE on the Minecraft tech-tree and find that it surpasses open-weight Voyager variants in both key tech-tree milestones and the number of unique items obtained. Section 5.5 further shows that when both agents begin with the same limited capabilities, a single round of GPT-4 seeding followed by just seven turns of peer-to-peer dialogue lifts the population-level success rate from 62% to 79%, revealing a Condorcet-style “many-minds” boost that stems from richer interaction rather than larger models. This generalization in post-collaborative settings paves the way for agents that can continuously adapt through social interaction, reducing the reliance on large-scale computing resources for training and fine-tuning the underlying language models.

2 Related Work

Theory of Mind in Embodied Agents Early attempts to equip machines with mental-state reasoning drew on deep learning [25–27], reinforcement learning [28], and Bayesian models [29]. These systems were either confined to toy 2-D worlds [25, 26] or failed to generalize to richer environments [27]. The emergence of large language models (LLMs) reignited interest in ToM, sparking evaluations of their putative capabilities [23, 30–32]. Although evidence so far suggests that LLMs lack genuine ToM, their breadth of knowledge has inspired a new wave of prompt-based frameworks [33–36]: the first two focus on perspective-taking, while the latter pair ground reasoning in explicit causal models. BIP-ALM remains the sole example that integrates such a framework with an LLM [37, 38].

LLM-based Multi-Agent Systems LLMs have demonstrated overwhelming success in performing a variety of single-agent tasks ranging from question answering and text generation to code synthesis and complex reasoning in embodied settings [11, 39, 40]. This success has started a growing interest in developing LLM-based multi-agent systems [21, 22, 41–43], where language-based communication is a central component. For instance, CoELA [21] introduce a modular multi-agent framework that integrates memory, perception and execution, while COMBO [41] present a multi-agent cooperative framework, that deals with collaboration in partial egocentric views of the environment. However, these frameworks lack an explicit belief representation and Theory-of-Mind reasoning. Recently, benchmarks like MineCollab [44], designed for collaborative reasoning, find that current LLM agents are not suited for multi-agent collaboration in long-horizon embodied scenarios, a gap that MINDFORGE addresses by endowing agents with structured perspective taking capabilities.

Single-agent Frameworks Recent research on long-horizon tasks in single-agent systems differs architecturally from MINDFORGE’s unified collaborative learning framework. For example, ExpeL [45] and CLIN [46] emphasize experiential learning through vectorized trajectories or causal abstractions but lack explicit belief graphs and integration across episodic, semantic, and procedural memories. Frameworks like DEPS [47] and ADaPT [48] enable in-episode self-correction via iterative refinement, yet insights are often ephemeral, confined to the context window without persistent storage or structured models of others’ beliefs. SSO [49] provides procedural memory through skill libraries but omits episodic/semantic layers and social reasoning for targeted teaching or belief revision. Similarly, Adam [50] structures world knowledge as causal graphs in open environments, while Optimus-1 [51] builds hierarchical semantic graphs from experiences.

3 Background

Voyager Voyager agents consist of three main components: (1) an automatic curriculum generator, (2) a skill library, and (3) an iterative prompting mechanism. The automatic curriculum proposes progressively complex tasks, driving continuous learning. The skill library stores and retrieves learned behaviors, facilitating knowledge accumulation and transfer. The iterative prompting mechanism generates and refines executable code based on environmental feedback, allowing for adaptive behavior. Voyager demonstrated significant capabilities in item discovery, technology progression, and skill generalization. This system established a new benchmark in open-world AI agents, showcasing the potential for lifelong learning and adaptation in complex, dynamic environments. The Voyager framework serves as a base for our lifelong *collaborative* learning agents within Minecraft.

BigToM Causal Template The BigToM causal template [23], provides a structured framework for representing Theory of Mind (ToM) scenarios as causal graphs. This template comprises key components including context, desire, percept, belief, causal event, and resulting actions. By representing ToM scenarios in this causal structure, the template allows for systematic manipulation of variables such as an agent’s perceptions, beliefs, and actions in response to environmental changes. This approach enables the creation of diverse, well-controlled ToM scenarios, capturing the complex causal relationships between an agent’s mental states and their behavior in social situations. The causal graph representation facilitates a more nuanced and systematic exploration of ToM reasoning, allowing for the examination of various inference patterns such as forward belief, forward action, and backward belief inferences in social cognitive tasks.

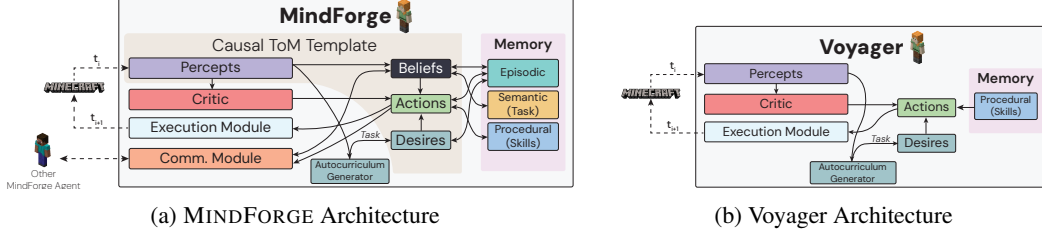


Figure 2: Cognitive architectures of MINDFORGE and Voyager respectively. The Voyager architecture has been re-framed within the scope of our cognitive architecture. MINDFORGE expands the original Voyager framework to enable communication through theory of mind together with additional memory subsystems to enhance the lifelong learning capabilities of the agent.

4 Method

To grant MINDFORGE agents the ability to learn through collaborative social interactions, we introduce the following components for communication and perspective-taking: (i) an **explicit structured representation** linking percepts, beliefs, desires, and actions (Section 4.1), (ii) **perspective taking** so that the agent can reason over self-beliefs and those about its collaborators (Section 4.2), (iii) a **communication medium** that the agents can use to share information in natural language (Section 4.3), and (iv) the ability to utilize different types of **memory subsystems** (Section 4.4).

4.1 Structured Representation of Own Mental States: Beliefs, Desires, and Intentions

For MINDFORGE agents to develop effective theory of mind capabilities, they must first possess structured representations of their own mental states. This explicit modeling of the self is crucial for *recursive social reasoning*: enabling agents not only to reason about their own beliefs, goals, perceptions, and how these might lead to their actions, but also to form a similar mental model for other agents. This mirrors findings in developmental psychology, where children first develop an awareness of their own thoughts, feelings, and goals before being able to infer the mental states of others [52, 53]. This is further supported by neuroscientific research that has also identified distinct neural mechanisms for self-cognition and social cognition [54, 55].

We draw on the classical Belief-Desire-Intention (BDI) framework [56] and operationalize it using the BigToM causal template (Section 3) [23]. In MINDFORGE, *beliefs* represent the agent’s current understanding of the world inferred from sensory perception, including interactions with other agents. *Desires* encode the agent’s goals, both generated by an automatic curriculum and those arising from collaborative interactions. *Intentions* correspond to concrete executable actions, implemented as generated code—and so, consistent with BigToM [23], we treat “intentions” and “actions” interchangeably in implementation. However, we note that this equivalence does not always hold conceptually: intentions refer to internal commitments or plans, while actions are observable behaviors that may diverge from intentions. For a deeper treatment of this distinction, see [57].

Belief Structure. We categorize a MINDFORGE agent’s beliefs into four key categories, illustrated in Figure 3: i. *perception-related beliefs*, formed from direct sensory input via the Minecraft API; ii. *task-related beliefs*, inferred from the agent’s reflection on its upcoming objectives; iii. *interaction-related beliefs*, derived from chat messages during task execution; and iv. *partner-related beliefs*, which track others’ mental states. Each belief type is generated via a separate LLM call, with prompts conditioned on current percepts, memory retrievals, and conversation history (see Appendix F). This modular belief formation feeds into the causal ToM template Figure 2a, which governs downstream action generation. Structured self-modeling thus serves as the foundation for cultural learning and perspective-taking.

4.2 Theory of Mind: Modeling Partner Perspectives

Building on the structured representation of their own mental states, MINDFORGE extend this modeling to others via Theory of Mind—the capacity to attribute beliefs, desires, and intentions to others. Each MINDFORGE agent achieves this by maintaining a partner-specific BigToM graph within its own beliefs, allowing it to reason about the internal state of each conversational partner

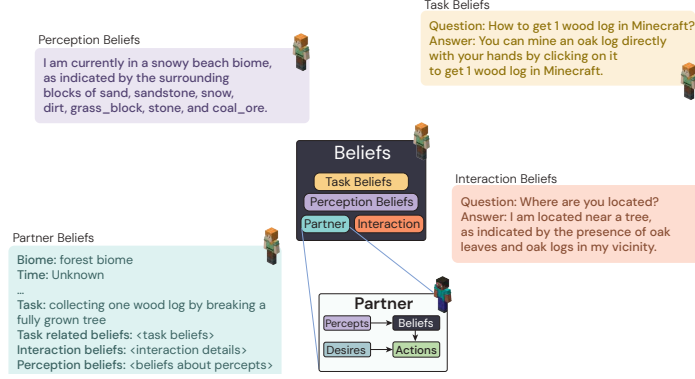


Figure 3: Each agent maintains a set of internal beliefs that it can exploit as needed. In order to enable ToM capabilities agents also maintain a belief over the beliefs, actions and desires of their collaborative counterparts.

(see Figure 3). These representations guide how an agent interprets and responds to incoming communication.

During collaboration, an agent uses its internal model of its partner to simulate what the partner knows, believes, or needs. The underlying LLM receives this partner model along with the conversation history to generate responses incorporating the collaborator’s point of view. As interaction proceeds, the partner beliefs are continuously updated using new received through the communication channel (Figure 6). This dynamic modeling of partner mental states endows MINDFORGE agents with one of the most crucial abilities within the cultural learning framework: reasoning about partner perspectives [14, 15]. For instance, an expert agent can diagnose a novice’s misconception and tailor its feedback accordingly, while a novice can ask more targeted questions with fewer digressions by simulating what the expert knows. See Appendix B.1 for more details on how perspective-taking is integrated into the communication module, and in-game examples.

4.3 Communication Medium

To take advantage of representations of each other’s mental models and collaborate effectively, agents need to be able to communicate with each other. We operationalize communication in natural language, integrating the MINDFORGE framework’s communication system with Minecraft’s native in-game chat functionality. Specifically, once a MINDFORGE agent fails to complete a task, it initiates a communication round with its partner where it leverages its BigToM representation to ask for help. This system enables agents to exchange textual information and, crucially, to integrate feedback from collaborators directly into their belief systems. Concretely, we utilize the Mineflayer API [58] to extract and process chat data from the Minecraft environment. By leveraging the game’s existing chat system, the framework ensures that collaborative interactions occur in a context-appropriate manner, closely mirroring human-to-human communication within Minecraft.

To handle the resulting conversation, we use a dedicated LLM instance within the MINDFORGE agent which is tasked with synthesizing and interpreting the conversation into the structured BigTOM BDI template, generating both interaction beliefs and partner beliefs (see prompts in Appendix F). This instance does not directly influence any other parts of the agent.

4.4 Memory Subsystems

A MINDFORGE agent’s memory consists of three subsystems drawing on research about human memory: (i) **episodic**, (ii) **semantic**, and (iii) **procedural** memory.

Episodic memory deals with pertinent information about the agent’s past attempts at a task and social interactions with other agents. Recollections of past episodes serve as context for humans to interpret others’ behaviors, leading to divergent future actions in the same situation in response to different recollections [59]. Humans also tend to summarize information from their social interactions, but at the same time are capable of preferentially accessing episodes when the credibility

of certain summaries is low due to the lack of evidence [60]. Moreover, findings also suggest that there are processes in the brain for updating these summaries as new episodes transpire [60, 61]. Inspired by complementary learning systems theory, Larimar [62] enhances LLMs with a distributed episodic memory module, enabling dynamic knowledge updates without any gradient-based learning. Furthermore, Optimus-1 integrates past experiences in a multimodal memory storage mechanism and abstracted experience pool used for summarization of past experiences. For a MINDFORGE agent, the role of episodic memory is twofold. Firstly, it stores details about its partners revealed during previous conversations that can be exploited for perspective-taking in future communication. Secondly, we store past actions that led to failure and retrieve them based on their similarity with the task at hand. Thus, a MINDFORGE agent can track both the progress of its partner and learn from past failures.

Semantic memory constitutes an individual’s store of knowledge about the world. The content of semantic memory is abstracted from the experience and is therefore said to hold summarized generalizations, likely without references to specific experiences [60, 61, 63]. For a MINDFORGE agent, the semantic memory stores information related to the task and environment. It stores information relevant to completing a task; for instance, the prerequisites for being able to mine dirt in the game world. Such a piece of information is not necessarily tied to one specific episode. Rather, it is generally useful information about the world and can be distilled from the episodic memory.

Finally, **procedural memory** stores the skills that the agent learns and is analogous to the skill library present in the original Voyager framework [11]. The skills stored in the procedural memory are reusable pieces of code that the agent can use for solving problems, allowing the agent to take on more complex tasks while avoiding re-learning elements previously used successfully. As illustrated in Figure 2a, episodic and semantic memory stores influence the agent’s beliefs, while procedural memory provides the routines that inform the agent’s next action. Additional information about the memory subsystems can be found in Appendix B.2, where we provide implementation details, and an ablation for the episodic memory component, where we show that omitting it from MINDFORGE leads to a drop in success rate.

5 Experiments

5.1 Experimental Setup

Our MINDFORGE framework introduces three advances that let agents improve continuously through social exchange: (1) explicit theory-of-mind representations for reasoning about peers, (2) Natural-language dialogue for coordination, and (3) integrated semantic and episodic memory to retain experience. We evaluate these additions in the same Minecraft world used by Voyager [11], built on MineDojo [64] with motor control via Mineflayer [58]. Component LLMs communicate through DSPy [65, 66] and LangChain [67] for generation and embedding. In the base experimental setting, we follow Voyager [11] and allow the agent to execute 4 sequential actions to solve a task. MINDFORGE agents that interact (see Section 5.3) interleave one communication round (via the in-game chat) between two environmental actions. A snapshot of this setup can be seen in Figure 4a. For isolated tasks we measure the fraction of successes over 24 independent trials (one task per trial). In tech-tree experiments we follow Voyager’s protocol, tracking milestone achievements and unique items collected. To ensure comparability, we preserve Voyager’s original prompt structure and workflow throughout.

5.2 Voyager With Open-Weight LLMs

To establish non-interactive performance baselines, we replace GPT-4 [68] with three open-weight LLMs in the Voyager framework [11]: Mistral-7B [69], Mixtral-8x7B [70], and Llama 3.1-8B [71], selected for their long context windows and manageable size. Testing focuses on two fundamental Minecraft tasks: collecting single blocks of dirt and wood. Results in Table 1 reveal that while GPT-4 achieves perfect performance, open-weight LLMs struggle significantly even with these basic tasks without collaboration. This aligns with Wang et al. [11]’s observation that GPT-4’s superior code generation capabilities enable it to obtain $5.7\times$ more unique items than other models, establishing a clear baseline for measuring the impact of our collaborative approach.

Table 1: Voyager [11] powered by open-weight LLMs (even fine-tuned for Minecraft) struggles with basic tasks. Closed-source models like GPT-4 exhibit perfect accuracy. A single MINDFORGE agent bridges the performance gap between Mistral-7B and GPT-4 on both wood and dirt collection, showing significant gains compared to other learning techniques such as PEFT.

Task	Mistral-7B	Mixtral-8x7B	Llama 3.1-8B	Mistral-7B (PEFT)	Mistral-7B (MindForge)	GPT-4
Dirt	7%	27%	4%	7%	37.5%	100%
Wood	4%	27%	7%	12.5%	33.3%	100%

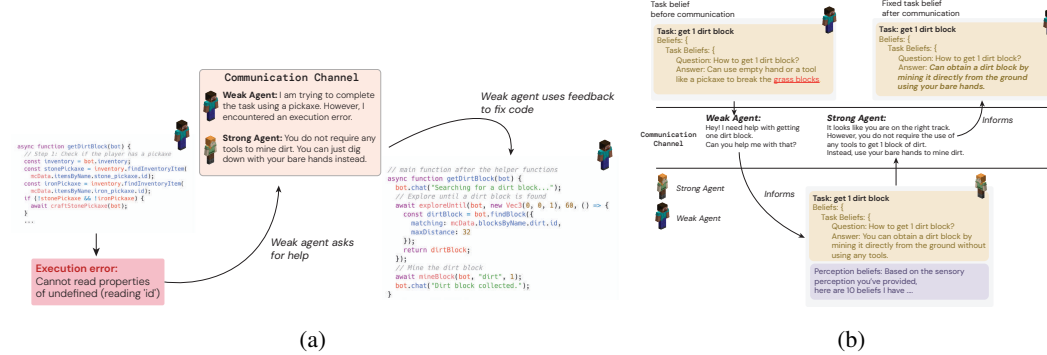


Figure 4: Failure mode correction. (a) Communication with an expert agent fixes the code error. (b) A weaker agent's incorrect task-related belief (initially believing it is mining 'grass blocks' and needs tools) is corrected by a stronger agent through communication (e.g., stating 'dirt blocks' do not require tools), leading to an updated task understanding.

Primary Failure Cases While Wang et al. [11] only discuss code-generation failures, we find two prominent failure cases for the open-weight Voyager setup: (1) **false beliefs** and (2) **faulty code generation**. Importantly, we discover that the critic-based self-improvement component in Voyager fails to recover from initial mistakes and subsequently fails to solve the task. As shown in Figure 4b, a Mistral-7B powered Voyager fails at dirt collection due to incorrect assumptions about tool requirements and lack of basic knowledge pertaining to Minecraft. Similarly, Figure 4a showcases how feedback from a stronger agent can solve code misconceptions about the Mineflayer API [58].

Fine-Tuning Does Not Help Parameter-efficient tuning often helps LLMs [72], yet adapting Mistral-7B with GPT-4 execution traces (reasoning + code) and Minecraft manuals (domain knowledge; Appendix E) produced no benefit. In cultural-learning terms [14, 15] this is imitation learning, but the tuned model still scores 7% on dirt collection (± 0) and 12.5% on wood collection ($+8.5$) as shown in Table 1. Post-hoc analysis shows the tuning makes outputs wordier, blurs decision-making, and erodes instruction-following, indicating that naively imitating unstructured traces is counter-productive.

5.3 Introducing Collaboration: MINDFORGE Agents

Having shown that PEFT-based imitation learning fails to improve agent capabilities, we evaluate our proposed MINDFORGE framework. We adopt an instructive learning approach within the cultural learning framework [15], where an expert agent guides a novice through communication and perspective-taking. This multi-agent collaboration effectively bridges the performance gap between open-weight LLMs and GPT-4, addressing the failure cases identified in Figure 4.

Feasibility: Human (Expert) Collaboration Improves Performance To establish feasibility, we first conduct a wizard-of-oz study where a human expert interacts with an open-weight LLM-powered MINDFORGE agent through the in-game chat. As shown in Table 2, human expert collaboration achieves on average a +57.85% increase in task completion rate for dirt collection over the non-

Table 2: Fraction of successful MINDFORGE agents across 24 individual trials; dirt and wood collection. Standard Voyager [11] setup: 4 attempts per task, with a communication round interleaved where appropriate. See also Figure 5 and Figure 7.

Setting	Task	Mistral-7B	Mistral-8x7B	Llama 3.1-8B
w/o interactions	Dirt	37.5%	29.15%	12.5%
instructive setting, human expert	Dirt	+45.83	+57.85	+45.83
instructive setting, GPT-4 expert	Dirt	+16.16	+37.51	+4.1
instructive setting, Llama 3.1-70B	Dirt	+8.33%	-	-
collaborative setting, no expert	Dirt	-4.2	+0	-8.5
w/o interactions	Wood	33.3%	50%	7%
instructive setting, GPT-4 expert	Wood	+12.52	+20.8	+5.5
instructive setting, Llama 3.1-70B	Wood	+8.36%	-	-
collaborative setting, no expert	Wood	-12.5	-4.15	+0

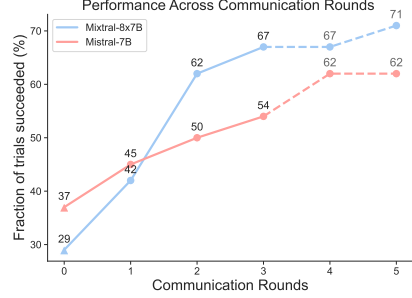


Figure 5: **Instructive Setting (GPT-4); Dirt Collection.** Performance scales with communication rounds. MINDFORGE w/o interactions as a baseline. Actions are interleaved with communication rounds. Dashed (—): extra collaboration rounds. $\triangle$ points: cold-start (4 actions) without interactions.

interactive baselines. This demonstrates both the viability and flexibility of our framework in enabling productive human-agent interactions for collaborative tasks.

GPT-4 (Expert) Collaboration Maintains Performance Benefits We replace the human expert with a GPT-4-powered MINDFORGE agent that collaborates with an open-weight LLM agent. The expert agent infers its partner’s failure cases through perspective-taking based on natural language communication in the game chat. Unlike human collaboration, agents cannot directly access each other’s internal beliefs and must rely on Theory of Mind through communication to model their counterpart’s mental state. This perspective-taking enables the expert to accurately diagnose and address the weaker agent’s misconceptions. Notably, this setup generalizes the classic notion of knowledge distillation beyond logits or model weights: here, distillation unfolds through social interaction, where the expert agent transmits structural knowledge, corrects false beliefs, and shapes the student’s internal representations in real time. As shown in Table 2, this approach significantly improves performance compared to the non-interactive baselines, achieving on average a 16.1% increase in task completion rates for dirt collection and a 13% increase for wood collection tasks respectively. A sample conversation can be found in Appendix D.

Ablations To elucidate the MINDFORGE framework’s components, we conducted targeted ablations. The perspective-taking module, enabling agents to reason over partners’ beliefs, boosts task completion by up to 13%, as evidenced by comparisons across multiple communication rounds (see Table 7, Appendix B.1). Our structured ToM representation outperforms its unstructured variant in complex Minecraft tasks (Table 8). Additionally, ablating the episodic memory component reveals that retaining past failures enhances performance (Table 9, Appendix B.2).

More Communication Rounds Improve Performance Since communication is central to our framework, we examine the benefits of multiple rounds per task trial, assessing communication and perspective-taking amid escalating opportunities. We pair a GPT-4 MINDFORGE expert with a Mistral-8x7B MINDFORGE weak agent. Figure 5 demonstrates communication’s value through a sharp rise in task completion rates with additional rounds, while Figure 6 illustrates the expert’s progressively accurate grasp of the partner’s mental state. Further evidence in Appendix A.1 affirms communication’s importance in instructive and collaborative settings, even for demanding Minecraft tasks (“Craft a pickaxe” and “Mine Iron”) involving MINDFORGE agents.

Using a less constrained communication protocol We introduce the MINDFORGE communication protocol in Section 4.3, initiating agent conversation only after task failure. To demonstrate the framework’s flexibility, we test two MINDFORGE tasks with a dynamic communication setup, allowing the weak agent to choose whether to initiate communication when uncertain or attempt the task independently. Table 6 (Appendix A.3) reveals comparable task completion rates between the

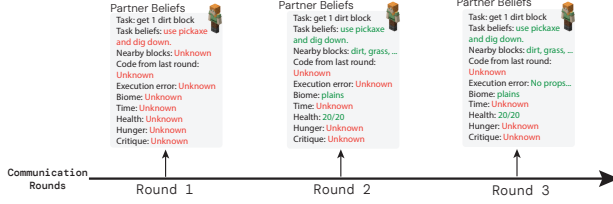


Figure 6: Partner beliefs of the strong agent progressively improve as the number of communication rounds increase. Subsequently, these beliefs are used to reason throughout the conversation and take perspective.

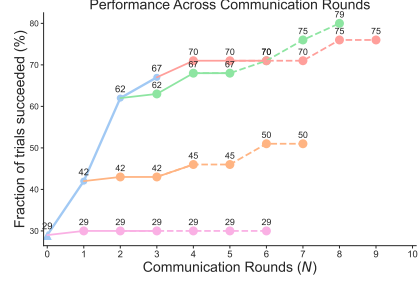


Figure 7: **Collaborative Setting (Mixtral); Dirt Collection.** Blue: GPT-4-guided baseline. Dashed (– –): extra collaboration rounds.

Table 3: We follow Voyager’s methodology [11] for reporting the Minecraft tech-tree results. The reported scores represent the number of prompting iterations until the milestone was accomplished, averaged over three trials. The fewer the iterations, the more efficient the method. *N/A* represents the agent being unable to solve the task. **Best performance** in terms of iterations required and number of times milestone is reached (out of 3 trials).

Method	Model	Wooden Tool	Stone Tool	Iron Tool
Voyager [2023]	GPT-4	6 ± 2 (3/3)	11 ± 2 (3/3)	21 ± 7 (3/3)
ReAct [2023]	GPT-4	N/A (0/3)	N/A (0/3)	N/A (0/3)
Reflexion [2023]	GPT-4	N/A (0/3)	N/A (0/3)	N/A (0/3)
AutoGPT [2023]	GPT-4	92 ± 72 (3/3)	92 ± 72 (3/3)	135 ± 103 (3/3)
Voyager (open-weight)	Mixtral 8 × 7B	N/A (0/3)	N/A (0/3)	N/A (0/3)
	Llama 3.1-8B	N/A (0/3)	N/A (0/3)	N/A (0/3)
	Llama 3.1-70B	19 ± 9 (3/3)	N/A (0/3)	N/A (0/3)
MINDFORGE (w/o collaboration)	Mixtral 8 × 7B	86 ± 17 (3/3)	N/A (0/3)	N/A (0/3)
	Llama 3.1-8B	120 ± 0 (1/3)	N/A (0/3)	N/A (0/3)
	Llama 3.1-70B	50 ± 17 (3/3)	113 ± 24 (2/3)	N/A (0/3)
MINDFORGE (mixed-expertise collaboration)	Mixtral 8 × 7B	51 ± 20 (3/3)	N/A (0/3)	N/A (0/3)
	Llama 3.1-8B	100 ± 30 (3/3)	N/A (0/3)	N/A (0/3)
	Llama 3.1-70B	62 ± 14 (3/3)	79 ± 25 (3/3)	120 ± 30 (2/3)
MINDFORGE (mixed-expertise collab w/ Llama 3.1-70B)	Mixtral 8 × 7B	65 ± 12 (3/3)	N/A (0/3)	N/A (0/3)

default and flexible protocols, underscoring MINDFORGE’s independence from specific communication designs. Qualitatively, weaker models like Mistral-7B consistently act confidently, never seeking help despite the option, while both models adeptly bypass assistance for tasks aligned with prior experience.

5.4 MINDFORGE Lifelong Learning

We evaluate MINDFORGE’s lifelong learning capabilities against both GPT-4 and open-weight Voyager [11] variants. The progression through increasingly complex technological achievements (from wooden to diamond tools) provides a rigorous test-bed for assessing lifelong learning and collaboration abilities of MINDFORGE. We follow the methodology from Voyager and evaluate performance on three key milestones: (1) *wooden tools*, (2) *stone tools* and (3) *iron tools*.

In line with the findings in Table 1, Table 3 highlights the performance gap between GPT-4 powered Voyager and its open-weight variants, with only Llama 3.1-70B [71] reaching the *wooden tools* milestone. On the other hand, MINDFORGE shows substantial improvement across the board both in terms of achieved milestones and efficiency (number of iterations). Notably, Llama 3.1-70B [71] powered MINDFORGE achieves two additional tech-tree milestones, nearing the performance with the original GPT-4 powered Voyager [11]. We observe a steady increase in efficiency (number of iterations) as the size of the LLM increases: Llama 3.1-70B [71] powered MINDFORGE is $1.62\times$ more efficient than its Llama 3.1-8B counterpart for reaching the *wooden tools* milestone, suggesting

that larger models can better utilize the collaborative learning framework. Furthermore, to understand the impact of collaboration in a mixed-expertise collaborative setting, we experiment with a non-collaborative MINDFORGE ablation. Notably, MINDFORGE agents without collaboration require on average 15 more prompting iterations to reach the *wooden tools* milestone and 34 iterations to reach the *stone tools* milestone. Thus, the collaborative framework not only enhances task completion rates but also enables more efficient acquisition of complex skills. We further test the generalization capabilities of MINDFORGE agents in Appendix A.2 where we show how even after collaboration finishes, the agents can reuse the knowledge in out-of-distribution tasks.

5.5 MINDFORGE Collaborative Learning Setting

While prior sections focused on instructive learning from stronger peers, we now consider a more challenging setup: collaboration between two MINDFORGE agents of **matched capabilities**, each powered by the same type of open-weight LLM. This setting reflects mutual cultural learning without access to an expert where the agents must bootstrap improvements solely through interaction, thereby mimicking certain qualities of emergence. Critically, we evaluate not individual trajectories, but instead focus on **population-level performance**: after N rounds of communication, what fraction of agent pairs from the total pool succeed in completing the task? As shown in Table 2, when agents begin from unprimed states, collaboration often leads to stagnation or even degradation which mirrors the “*blind leading the blind*” dynamic, where flawed mental models are mutually reinforced.

However, even minimal prior expert instruction—such as a single round of GPT-4-guided collaboration—can prime agents for more effective peer interaction, even if they still fail when acting alone. As shown in Figure 7, we pair Mixtral-8x7B agents that each received N communication rounds of GPT-4 guided trial before interacting. We observe that the fraction of successful agents steadily increases with additional communication rounds between themselves (Mixtral agents). From a 62% baseline population succeeding, we reach 79% of the population succeeding after seven turns, showing a 17% increment. This result reflects a form of test-time compute scaling, where richer interaction, not larger models, drives the emergent improvements across the group.

This also aligns with the Condorcet Jury Theorem [73], which predicts that collective decisions improve when individuals exceed a baseline level of competence. Here, we see a distributed variant of this principle in action: once agents cross a minimal competence threshold, the communication induced bootstrapping can yield reliable gains at the population level.

6 Conclusion

The MINDFORGE framework demonstrates that cultural learning can overcome the limitations of open-weight LLMs in embodied environments. Integrating theory of mind, natural communication, and structured memory systems enables MINDFORGE agents to learn from both mixed and matched expertise peers and overcome fundamental failure modes in perception and code generation. Our experiments show that collaboration significantly improves performance on lifelong learning scenarios, with agents collecting $2.3\times$ more unique items and achieving $3\times$ more tech-tree milestones when paired with expert agents. Crucially, in the collaborative setting, we show that the fraction of successful agents scales as a function of communication rounds; revealing emergent gains through perspective taking and interaction. These results suggest that social learning could reduce embodied AI’s dependence on large proprietary models, opening new paths for democratized agent development aligned with Green AI principles [74].

Acknowledgments and Disclosure of Funding

We thank OpenAI’s Researcher Access Program for providing API credits that enabled more comprehensive experiments. We are also grateful to Dr. Max van Duijn for his constructive critique and insights on Theory of Mind, to Dr. Jesse Krijthe for his thoughtful input on the Condorcet Jury Theorem, and to members of the Pattern Recognition Lab at TU Delft for their valuable feedback.

References

- [1] Eric Kolve, Roozbeh Mottaghi, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.
- [2] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9339–9347, 2019.
- [3] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8494–8502, 2018.
- [4] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10740–10749, 2020.
- [5] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [6] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.
- [7] Maryam Zare, Parham M Kebria, Abbas Khosravi, and Saeid Nahavandi. A survey of imitation learning: Algorithms, recent developments, and challenges. *IEEE Transactions on Cybernetics*, 2024.
- [8] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International conference on machine learning*, pages 1282–1289. PMLR, 2019.
- [9] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.
- [10] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *arXiv preprint arXiv:2302.00487*, 2023.
- [11] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.
- [12] Nicholas K Humphrey. The social function of intellect. *Growing points in ethology*, pages 303–317, 1976.
- [13] Robin IM Dunbar. The social brain hypothesis. *Evolutionary Anthropology: Issues, News, and Reviews*, 6(5):178–190, 1998.
- [14] Michael Tomasello, Ann Cale Kruger, and Hilary Horn Ratner. Cultural learning. *Behavioral and brain sciences*, 16(3):495–511, 1993.
- [15] Michael Tomasello. Cultural learning redux. *Child development*, 87(3):643–653, 2016.
- [16] David Premack and Guy Woodruff. Does the chimpanzee have a theory of mind? *Behavioral and brain sciences*, 1(4):515–526, 1978.
- [17] Simon Baron-Cohen. The evolution of a theory of mind. In Michael C Corballis, editor, *The descent of mind: Psychological perspectives on hominid evolution* (pp, volume 361, pages 261–277. Oxford University Press, xii, 1999. URL <https://psycnet.apa.org/record/1999-04113-013>.
- [18] Marie Devaine, Guillaume Hollard, and Jean Daunizeau. The social bayesian brain: does mentalizing make a difference when we learn? *PLoS computational biology*, 10(12):e1003992, 2014.

- [19] Geoffrey Hinton. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
- [20] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.
- [21] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models. *arXiv preprint arXiv:2307.02485*, 2023.
- [22] Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023.
- [23] Kanishk Gandhi, Jan-Philipp Fränken, Tobias Gerstenberg, and Noah Goodman. Understanding social reasoning in language models with language models. *Advances in Neural Information Processing Systems*, 36, 2024.
- [24] John E Laird. *The Soar cognitive architecture*. MIT press, 2019.
- [25] Neil Rabinowitz, Frank Perbet, Francis Song, Chiyuan Zhang, SM Ali Eslami, and Matthew Botvinick. Machine theory of mind. In *International conference on machine learning*, pages 4218–4227. PMLR, 2018.
- [26] Yun-Shiuan Chuang, Hsin-Yi Hung, Edwinn Gamborino, Joshua Oon Soo Goh, Tsung-Ren Huang, Yu-Ling Chang, Su-Ling Yeh, and Li-Chen Fu. Using machine theory of mind to learn agent social network structures from observed interactive behaviors with targets. In *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1013–1019. IEEE, 2020.
- [27] Tianmin Shu, Abhishek Bhandwaldar, Chuang Gan, Kevin Smith, Shari Liu, Dan Gutfreund, Elizabeth Spelke, Joshua Tenenbaum, and Tomer Ullman. Agent: A benchmark for core psychological reasoning. In *International conference on machine learning*, pages 9614–9625. PMLR, 2021.
- [28] Fabio Grassiotto and Paula Dornhofer Paro Costa. Cogtom: A cognitive architecture implementation of the theory of mind. In *ICAART (2)*, pages 546–553, 2021.
- [29] Chris Baker, Rebecca Saxe, and Joshua Tenenbaum. Bayesian theory of mind: Modeling joint belief-desire attribution. In *Proceedings of the annual meeting of the cognitive science society*, volume 33, 2011.
- [30] Michal Kosinski. Theory of mind might have spontaneously emerged in large language models. *arXiv preprint arXiv:2302.02083*, 2023.
- [31] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.
- [32] Tomer Ullman. Large language models fail on trivial alterations to theory-of-mind tasks. *arXiv preprint arXiv:2302.08399*, 2023.
- [33] Alex Wilf, Sihyun Shawn Lee, Paul Pu Liang, and Louis-Philippe Morency. Think twice: Perspective-taking improves large language models’ theory-of-mind capabilities. *arXiv preprint arXiv:2311.10227*, 2023.
- [34] Pei Zhou, Aman Madaan, Srividya Pranavi Potharaju, Aditya Gupta, Kevin R McKee, Ari Holtzman, Jay Pujara, Xiang Ren, Swaroop Mishra, Aida Nematzadeh, et al. How far are large language models from agents with theory-of-mind? *arXiv preprint arXiv:2310.03051*, 2023.
- [35] Melanie Sclar, Sachin Kumar, Peter West, Alane Suhr, Yejin Choi, and Yulia Tsvetkov. Minding language models’ (lack of) theory of mind: A plug-and-play multi-character belief tracker. *arXiv preprint arXiv:2306.00924*, 2023.

- [36] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhit-ing Hu. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*, 2023.
- [37] Chuanyang Jin, Yutong Wu, Jing Cao, Jiannan Xiang, Yen-Ling Kuo, Zhiting Hu, Tomer Ullman, Antonio Torralba, Joshua B Tenenbaum, and Tianmin Shu. Mmtom-qa: Multimodal theory of mind question answering. *arXiv preprint arXiv:2401.08743*, 2024.
- [38] Huao Li, Yu Quan Chong, Simon Stepputtis, Joseph Campbell, Dana Hughes, Michael Lewis, and Katia Sycara. Theory of mind for multi-agent collaboration via large language models. *arXiv preprint arXiv:2310.10701*, 2023.
- [39] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.
- [40] Kevin Lin, Christopher Agia, Toki Migimatsu, Marco Pavone, and Jeannette Bohg. Text2motion: From natural language instructions to feasible plans. *Autonomous Robots*, 47(8):1345–1365, 2023.
- [41] Hongxin Zhang, Zeyuan Wang, Qiushi Lyu, Zheyuan Zhang, Sunli Chen, Tianmin Shu, Yilun Du, and Chuang Gan. Combo: Compositional world models for embodied multi-agent cooperation. *arXiv preprint arXiv:2404.10775*, 2024.
- [42] Yashar Talebirad and Amirhossein Nadiri. Multi-agent collaboration: Harnessing the power of intelligent llm agents. *arXiv preprint arXiv:2306.03314*, 2023.
- [43] Alexander Sasha Vezhnevets, John P Agapiou, Avia Aharon, Ron Ziv, Jayd Matyas, Edgar A Duéñez-Guzmán, William A Cunningham, Simon Osindero, Danny Karmon, and Joel Z Leibo. Generative agent-based modeling with actions grounded in physical, social, or digital space using concordia. *arXiv preprint arXiv:2312.03664*, 2023.
- [44] Isadora White, Kolby Nottingham, Ayush Maniar, Max Robinson, Hansen Lillemark, Mehul Maheshwari, Lianhui Qin, and Prithviraj Ammanabrolu. Collaborating action by action: A multi-agent llm framework for embodied reasoning. *arXiv preprint arXiv:2504.17950*, 2025.
- [45] Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19632–19642, 2024.
- [46] Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Peter Jansen, Oyvind Tafjord, Niket Tandon, Li Zhang, Chris Callison-Burch, and Peter Clark. Clin: A continually learning language agent for rapid task adaptation and generalization. *arXiv preprint arXiv:2310.10134*, 2023.
- [47] Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojuan Ma, and Yitao Liang. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*, 2023.
- [48] Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. Adapt: As-needed decomposition and planning with language models. *arXiv preprint arXiv:2311.05772*, 2023.
- [49] Kolby Nottingham, Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Sameer Singh, Peter Clark, and Roy Fox. Skill set optimization: Reinforcing language model behavior via transferable skills. *arXiv preprint arXiv:2402.03244*, 2024.
- [50] Shu Yu and Chaochao Lu. Adam: An embodied causal agent in open-world environments. *arXiv preprint arXiv:2410.22194*, 2024.
- [51] Zaijing Li, Yuquan Xie, Rui Shao, Gongwei Chen, Dongmei Jiang, and Liqiang Nie. Optimus-1: Hybrid multimodal memory empowered agents excel in long-horizon tasks. *arXiv preprint arXiv:2408.03615*, 2024.

- [52] H M Wellman and J D Woolley. From simple desires to ordinary beliefs: the early development of everyday psychology. *Cognition*, 35(3):245–275, 1 jun 1990. doi: 10.1016/0010-0277(90)90024-e. URL [http://dx.doi.org/10.1016/0010-0277\(90\)90024-E](http://dx.doi.org/10.1016/0010-0277(90)90024-E).
- [53] Josef Perner. *Understanding the Representational Mind*. The MIT Press, 03 1993. ISBN 9780262316033. doi: 10.7551/mitpress/6988.001.0001. URL <https://doi.org/10.7551/mitpress/6988.001.0001>.
- [54] Kevin N Ochsner, Rebecca D Ray, Jeffrey C Cooper, Elaine R Robertson, Sita Chopra, John D E Gabrieli, and James J Gross. For better or for worse: neural systems supporting the cognitive down- and up-regulation of negative emotion. *Neuroimage*, 23(2):483–499, 1 oct 2004. doi: 10.1016/j.neuroimage.2004.06.030. URL <http://dx.doi.org/10.1016/j.neuroimage.2004.06.030>.
- [55] R Saxe and N Kanwisher. People thinking about thinking people. the role of the temporo-parietal junction in “theory of mind”. *Neuroimage*, 19(4):1835–1842, August 2003. doi: 10.1016/s1053-8119(03)00230-1. URL <https://pubmed.ncbi.nlm.nih.gov/12948738/>.
- [56] Anand S Rao, Michael P Georgeff, et al. Bdi agents: from theory to practice. In *Icmas*, volume 95, pages 312–319, 1995.
- [57] Hayley Hung, Litian Li, Jord Molhoek, and Jing Zhou. The discontent with intent estimation in-the-wild: The case for unrealized intentions. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–9, 2024.
- [58] PrismarineJS. Mineflayer, 2021. URL <https://prismarinejs.github.io/mineflayer/>.
- [59] Aaron M Bornstein, Mel W Khaw, Daphna Shohamy, and Nathaniel D Daw. Reminders of past choices bias decisions for reward in humans. *Nature Communications*, 8(1):15958, 2017.
- [60] S Klein, L Cosmides, C Gangi, Betsy Jackson, J Tooby, and Kristi A Costabile. Evolution and episodic memory: An analysis and demonstration of a social function of episodic recollection. *Soc. Cogn.*, 27(2):283–319, 21 apr 2009. doi: 10.1521/SOCO.2009.27.2.283. URL <http://guilfordjournals.com/doi/10.1521/soco.2009.27.2.283>.
- [61] Stanley B Klein, Judith Loftus, and John F Kihlstrom. Self-knowledge of an amnesic patient: Toward a neuropsychology of personality and social psychology. *b; Tulving*, 1983:1989, 1993.
- [62] Payel Das, Subhajit Chaudhury, Elliot Nelson, Igor Melnyk, Sarath Swaminathan, Sihui Dai, Aurélie Lozano, Georgios Kollias, Vijil Chenthamarakshan, Soham Dan, et al. Larimar: Large language models with episodic memory control. *arXiv preprint arXiv:2403.11901*, 2024.
- [63] Jeffrey R Binder and Rutvik H Desai. The neurobiology of semantic memory. *Trends Cogn. Sci.*, 15(11):527–536, 1 nov 2011. doi: 10.1016/j.tics.2011.10.001. URL <http://dx.doi.org/10.1016/j.tics.2011.10.001>.
- [64] Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *Advances in Neural Information Processing Systems*, 35:18343–18362, 2022.
- [65] Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. Dspy: Compiling declarative language model calls into self-improving pipelines. 2024.
- [66] Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive NLP. *arXiv preprint arXiv:2212.14024*, 2022.
- [67] LangChain. Langchain documentation, 2024. URL <https://python.langchain.com/>. Accessed: 2024-11-19.

- [68] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [69] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. Mistral 7b, 2023. URL <https://arxiv.org/abs/2310.06825>.
- [70] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.
- [71] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.
- [72] Zeyu Han, Chao Gao, Jinyang Liu, Sai Qian Zhang, et al. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*, 2024.
- [73] David Austen-Smith and Jeffrey S Banks. Information aggregation, rationality, and the condorcet jury theorem. *American political science review*, 90(1):34–45, 1996.
- [74] Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. Green AI, August 2019.
- [75] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: We back up the claims made in the abstract and introduction in Section 5, where we run extensive experiments in Minecraft and show the benefits of the proposed framework. Furthermore, we offer qualitative results through various figures and text snippets in the manuscript. These are available in the main paper as well as in the appendix (e.g. Appendix D).

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[No\]](#)

Justification: We acknowledge the limitations of MINDFORGE and we discuss them throughout the paper when necessary. That said we do not have a standalone Limitations section.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The manuscript does not include any theoretical result.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Since our method relies on LLMs, we provide a full comprehensive list of the models used (via API), the underlying prompts used to query the LLM as well as extensive details in Section 5 that allows for potential reproducibility. Importantly, our method uses significant scaffolding to enable multi-agent communication in Minecraft. While we do not include code snippets in the manuscript, we provide the full code as part of the Supplementary Materials.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

- (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the full code repository used to execute any experiment referenced in the manuscript in the Supplementary Material. We anonymize the code and provide adequate instructions inside the README for reproducibility.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Since the method to interact with LLMs is through text prompts, we provide these artifacts together with how we set up the experiments. These details can be found in Section 5. The full details (code) are provided in the Supplementary Material.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report error bars where possible throughout this manuscript. Specifically, we provide error bars for the Minecraft tech-tree experiments in Table 3. Moreover, to ensure proper statistical significance, we perform a large number of individual trials across all the experiments in Section 5.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: While our paper is not based on direct compute resources like GPUs, we do acknowledge in Appendix G the cloud providers used throughout the experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: We have followed the NeurIPS Code of Ethics while writing and working at this manuscript.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [No]

Justification: While the MINDFORGE framework is designed to work with both humans and agents, we believe there is no foreseeable negative societal impact that can stem from this specific manuscript. However, we do mention that our framework might decrease the dependence on proprietary models and thus affect in a positive way global problems like climate change.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our paper does not pose such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We credit the LLM providers in Appendix G both for crediting their work and helping the reader reproduce the results in the paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: Since MINDFORGE makes use of existing open-weight LLMs, our paper does not release any new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The manuscript and research process does not involve crowdsourcing nor research with human subjects. While Table 2 showcases a feasibility study with a human expert, the human is one of the authors.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The manuscript and research process does not involve crowdsourcing nor research with human subjects. While Table 2 showcases a feasibility study with a human expert, the human is one of the authors.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We do not use LLMs to formulate our research problem however LLMs are being used for the agents that we are proposing.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

MindForge: Empowering Embodied Agents with Theory of Mind for Lifelong Cultural Learning

Appendices

A Additional Experimental Results

A.1 Collaborative Experiments

We provide an extended set of experiments to reinforce our findings in Figure 5 and Figure 7. Specifically, we consider atomic tasks that are significantly harder than dirt and wood collection in order to demonstrate the generality of our framework: (1) mining an iron block and (2) crafting a wooden pickaxe.

Table 4: Agent performance increases as the number of rounds grows in both collaborative and instructive settings. In the instructive setting, we use Llama 3.1-70B as the weak agent and GPT-4 as the strong agent. Similarly, collaborative experiments use Llama 3.1-70B for both MINDFORGE agents.

Setting	Task	Comm. Round 0	Comm. Round 1	Comm. Round 2	Comm. Round 3
Instructive	Mine Iron	41%	50%	54%	62%
Instructive	Craft a Pickaxe	20%	33%	41%	45%
Collaborative	Craft a Pickaxe	20%	25%	33%	33%

As Table 4 shows, extend communication and perspective-taking between MINDFORGE agents enable significant improvements on complex Minecraft tasks that require compositionality (41% $\rightarrow$ 62% and 20% $\rightarrow$ 45%). Similarly to Figure 7, we see that a purely collaborative setting results in smaller performance gains (20% $\rightarrow$ 33%) than instructive learning, due to reinforcing wrong beliefs and similar thinking patterns.

A.2 Post-Collaboration and OOD tasks

Do social interactions have long-term effects on agents and help them perform better on different tasks? We consider the impact of interactions in post-collaboration and out-of-distribution scenarios to assess MINDFORGE’s generalization and lifelong learning capabilities. Specifically, we define post-collaboration scenarios as a non-collaborative setting where the MINDFORGE agent tries to complete a task based on prior beliefs formed from a previous collaborative experience stored in the semantic memory. Moreover, we consider scenarios where the agent is tasked with solving a novel task with prior collaborative experience on an adjacent task in the Minecraft tech-tree. As Table 5 shows, collaboration between MINDFORGE agents together with semantic memory showcase an average improvement of 8.33% for in-distribution tasks (dirt collection) together with an average improvement of 8.78% in non-collaborative out-of-distribution scenarios (wood collection). These results showcase the importance of semantic memory in achieving continual learning through collaboration.

Table 5: Post-collaboration and out-of-distribution generalization of the MINDFORGE agent. We report the task completion rates across 24 individual trials.

Scenario	Task	Mistral-7B	Mixtral-8x7B	Llama 3.1-8B
No Collaboration	Dirt	37.5%	29.15%	12.5%
Post-Collaboration	Dirt	37.5%	41.66%	25%
No Collaboration	Wood	33.3%	50%	7%
OOD post-collaboration	Wood	41.66%	58.33%	16.66%

A.3 Alternative Communication Protocols

Table 6 shows the performance of MINDFORGE agents when they are allowed to begin a conversation at point compared to when they can only initiate a conversation upon failure.

Table 6: Comparison between the default MINDFORGE communication protocol and a more flexible alternative where agents can initiate the conversation at any point. Both protocols achieve similar results across different model sizes.

Model	Task	Setting	Task Completion Rate			
			Round 0	Round 1	Round 2	Round 3
Mixtral-8x7B	Mine dirt	MindForge w/ flexible communication	37%	45%	67%	67%
		MindForge	29%	42%	62%	67%
	Mine dirt and wood	MindForge w/ flexible communication	75%	79%	79%	83%
		MindForge	75%	79%	79%	83%
Mistral-7B	Mine dirt	MindForge w/ flexible communication	37%	42%	45%	54%
		MindForge	37%	42%	45%	54%
	Mine dirt and wood	MindForge w/ flexible communication	41%	45%	50%	50%
		MindForge	41%	45%	50%	50%

B Additional Ablations

B.1 Perspective Taking

To quantitatively assess the effect of perspective-taking, we perform an ablation study where we attempt to solve a Minecraft task without perspective-taking during communication in an instructive learning setting. Table 7 showcases how the absence of perspective-taking leads to worse performance, irrespective of how much agents communicate. Moreover, as the agents use more communication rounds when trying to solve a task, taking perspective of the other’s agent situation increases task-completion rate.

Table 7: Perspective-taking ablation. We consider the Minecraft task of collecting a dirt block. Improvement is quantified as the fraction of agents that solve the task across 24 individual trials.

Model Variant	Task Completion Rate			
	Round 0	Round 1	Round 2	Round 3
MINDFORGE w/ perspective-taking	29%	42%	61%	67%
MINDFORGE w/o perspective-taking	29%	37%	50%	54%
Improvement	0%	+5%	+11%	+13%

We attribute the positive correlation between performance and usage of perspective-taking during communication to the ability of the MINDFORGE agent to provide more relevant and directed advice. This observation is also supported by the substantial increase in performance in the first two rounds as presented in Table 7, where the teacher agent provides more insightful information as it gets to know the other’s agent situation better.

Additionally, we further ablate our structured representation of Theory of Mind by considering an unstructured perspective-taking 2-step prompt presented in Think Twice [33]. Table 8 showcases that a structured ToM representation provides a meaningful advantage on complex Minecraft tasks.

Table 8: Structured perspective-taking ablation with unstructured alternative. The unstructured ToM approach is inspired from Think Twice [33]. Using a structured belief-system results in higher performance on complex Minecraft tasks.

Task	Model Variant	Task Completion Rate			
		Round 0	Round 1	Round 2	Round 3
Craft a Pickaxe	MINDFORGE	20%	33%	41%	45%
Craft a Pickaxe	MINDFORGE w/o structured ToM	20%	29%	33%	41%
Mine Iron	MINDFORGE	41%	50%	54%	62%
Mine Iron	MINDFORGE w/o structured ToM	41%	50%	54%	58%

B.2 Memory Components

MINDFORGE employs three types of memory subsystems, as discussed in Section 4.4. These components ensure that the agent can reuse and recall relevant information about the task it is trying to complete from past experiences and interactions with other agents. The importance of procedural memory with respect to task completion is already studied in [11], concluding that reusing previous code snippets becomes increasingly important with the difficulty of the tasks. To understand the effect of episodic memory in the MINDFORGE framework, we perform an ablation on the episodic memory component within two Minecraft tasks. Table 9 showcases the drop in completion rate without the use of the episodic memory component. While we do not consider the tech-tree experiments here, we expect that considering past failure episodes leads to increasingly better results.

Table 9: Ablation on the episodic memory component in MINDFORGE on a single task. We report the task completion rates across 24 individual trials.

Scenario	Task	Mixtral-8x7B	Llama 3.1-8B
MINDFORGE w/ episodic memory	Dirt	29.15%	12.5%
MINDFORGE w/o episodic memory	Dirt	25%	16.66%
MINDFORGE w/ episodic memory	Wood	50%	7%
MINDFORGE w/o episodic memory	Wood	45.83%	7%

C Implementation details

C.1 Perspective Taking

Perspective-taking represents a central part of our framework since it allows MINDFORGE agents to take advantage of the internal causal structure of other agents in order to aid more efficient and grounded communication. In practice, perspective-taking corresponds to an LLM call where the prompt contains instructions and relevant information required to understand the internal state of other MINDFORGE agents during communication. Figure 8 showcases how perspective-taking integrates with the overall communication mechanism.

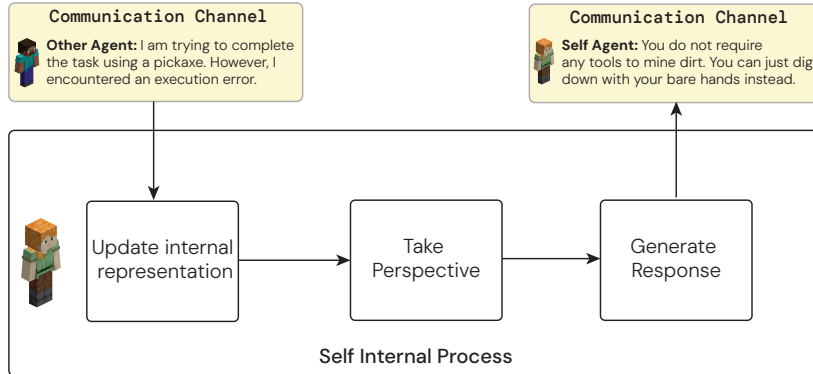


Figure 8: Flow of MINDFORGE communication components used to generate a natural language response. The agents’ internal process consists of two stages: (i) internal representation update and (ii) perspective-taking. Leveraging both processes the agent can then generate a natural language response and send it through the Minecraft chat.

Once the agent receives a message in the Minecraft chat, a first LLM call is executed where the agent updates the stateful internal representation about the agent that sent the message. This update process is shown in Figure 6. Subsequently, the updated internal representation is used as context together with the rest of the conversation to take perspective of the situation of the other MINDFORGE agent. Similar to the first stage, perspective-taking is implemented as a LLM call, with the following prompt:

Prompt for perspective-taking

You are a Minecraft agent named {{name}} and you are having a conversation with another agent named {{other_name}}.

Based on the current conversation and your knowledge about the other agent, {{other_name}}, take the other agent's perspective to assess and describe your current understanding, knowledge state, and likely needs from {{other_name}}'s perspective.

Here is the current conversation between you and {{other_name}}: {conversation}

Here is your mental model of {{other_name}}: {world_model}

Perspective Analysis:

Below you can find an example of perspective-taking from Minecraft gameplay:

Expert agent taking perspective of weak agent

Weak's current knowledge state includes understanding the biome they are in, the time of day, and the fact that there are dark oak logs nearby. Weak knows that a wooden axe is necessary for mining logs, and that crafting this tool will be a required step before proceeding with the task. Weak's likely needs from Strong include further assistance in navigating the environment during the night to find a tree or waiting until the day.

C.2 Episodic Memory

We implement episodic memory as a Retrieval-Augmented Generation (RAG) using LangChain [67]. Specifically, we embed all the episodes where the agent failed to complete the task using **text-embedding-ada-002**. An episode consists of the context used to generate the action, the code itself and the corresponding critic message. Subsequently, when the agent generates a new action, the RAG is queried to retrieve the **k=5** most relevant failure episodes with respect to the task at hand. Lastly, to ensure we do not populate the context window with too many tokens, we generate a summary (using an LLM) of the retrieved episodes and feed this summary in the context used to generate the action. Below you can find the corresponding prompts for generating the episodic summary:

System message

You are a helpful assistant tasked with summarizing past experience episodes and pointing out the causes of failure. Create a concise summary.

Prompt message

Please summarize these episodes and why they failed: {combined_episodes}

C.3 Procedural Memory

The procedural memory has an identical design and functionality with the skill library introduced in Voyager [11]. Once a task gets successfully completed, we store the corresponding piece of code such that we can later reference it in adjacent tasks. This becomes increasingly important in tasks that require multiple steps, like crafting a pickaxe: the agent needs to first create a crafting table, create wooden sticks and only then attempt to craft the pickaxe. When we run the MINDFORGE framework in an open-ended setting, the procedural memory allows the agent to continuously evolve and create an increasingly large collection of reusable skills.

C.4 Semantic Memory

As described in Section 4.4, semantic memory stores certain beliefs the agent has about the environment, coming from experience with the environment and the other MINDFORGE agents. Whenever MINDFORGE attempts a new task, it creates a certain belief about how the task should be solved. For example, the agent might believe it requires tools to mine a block of dirt. This belief is then added to the context of the LLM when the environment action is generated. As highlighted in Section 1, one of the primary failure cases of Voyager are the false beliefs that come from this LLM generation. However, compared to Voyager [11], we leverage the communication with other MINDFORGE agents such that these beliefs get updated and refined. Below we showcase an example of false belief correction when trying to collect a block of wood:

Initial belief inside the semantic memory

```
{ "How to mine 1 wood log in Minecraft?": "Answer: To mine 1 wood log in Minecraft, you need to use an axe." }
```

Updated belief after communication

```
{ "How to mine 1 wood log in Minecraft?": "Answer: To mine 1 wood log in Minecraft, you need to punch a tree with your bare hands." }
```

In addition to task-related memory, the MINDFORGE agent maintains a stateful set of beliefs about the other agents in the environment that are constantly updated across episodes. An example of this is presented in Figure 6, where the agent updates its representation based on the contents of the conversation. Thus, keeping track of various information and facts about other MINDFORGE agents acts as a semantic memory retrieval when agents communicate.

D Sample Conversation

Each communication round between two MINDFORGE agents comprises of six sending messages in a turn-based manner through the Minecraft build-in chat. Below you can find a sample conversation showcasing how the expert agent (Strong) helps the weak (Weak) agent.

Conversation about completing the task: "Mine 1 wood log"

Weak: Hey, can you help me with Mine 1 wood log?

Strong: Absolutely, I can help you with that. Given your current location in the dark forest biome and the presence of dark oak logs around you, you should be able to mine a log without any issues. Here's the code you can use: `'mine('dark_oak_log')`. This command will make your agent mine a dark oak log. Just make sure you are close enough to the tree when executing this.

Weak: Thank you for the helpful guidance, Strong! However, in my case, I am currently in a dark forest biome as well, but I am having trouble finding a tree during the night. Could you suggest a way to find a tree during the day so I can mine the wood log? I also noticed that I don't have a wooden axe in my inventory, so I may need to craft one before attempting to mine the log. I appreciate your assistance in this matter.

Strong: Of course, I understand your predicament. Since it's currently night time, visibility might be low which makes it challenging to locate a tree. You can wait until daybreak for better visibility. Use the `'wait_until_day()'` function to wait until it's daytime. For crafting a wooden axe, you would need 3 wood planks and 2 sticks. You can obtain wood planks from a wood log by using the `'craft('wooden_plank', wood_log)'` command and sticks from 2 wooden planks using the `'craft('stick', wooden_plank)'` command. Once you have these, you can craft a wooden axe using the `'craft('wooden_axe', wooden_plank, stick)'` function. Here's the code:

```
'''  
wait_until_day()  
craft('wooden_plank', wood_log)  
craft('stick', wooden_plank)  
craft('wooden_axe', wooden_plank, stick)  
'''
```

Once you have your axe and it's day time, you should be able to easily locate and mine a tree for the wood log.

Weak: Thank you, Strong. I appreciate your help.

E Fine-tuning setup

E.1 Execution Logs

For the first fine-tuning variant, we use execution logs generated by a Voyager agent powered by GPT-4. Specifically, these logs contain input and output text information obtained from the components specific to the architecture, such as the actor or the critic. We find this to be a meaningful way to transfer knowledge as well as reasoning capabilities from GPT-4 to Mistral-7B. To perform the fine-tuning, we use LoRA [75] due to its low computational requirements.

E.2 Wiki Data and Documentation

The second fine-tuning variation we consider builds upon the initial setup presented above and incorporates more structured information about Minecraft by including the Minecraft Wiki [64] and Minecraft API documentation. Similar to the first variant we use LoRA [75].

F Prompts

Prompt for updating partner beliefs

You are a Minecraft agent.

You just had a conversation with another agent based on a task you are trying to solve.

Based on the contents of the conversation and the previous beliefs, you have to create a set of beliefs that represent your perception of the other agent.

Prompt for creating interaction beliefs

You are a Minecraft agent.

You just had a conversation with another agent based on a task you are trying to solve.

Based on the contents of the conversation and the previous beliefs, you have to create a set of beliefs that that can help you complete the task.

Prompt for perspective-taking

You are a Minecraft agent named {{name}} and you are having a conversation with another agent named {{other_name}}.

Based on the current conversation and your knowledge about the other agent, {{other_name}}, take the other agent’s perspective to assess and describe your current understanding, knowledge state, and likely needs from {{other_name}}’s perspective.

Here is the current conversation between you and {{other_name}}: {conversation}

Here is your mental model of {{other_name}}: {world_model}

Perspective Analysis:

G LLMs

This section details the technical specifications of the Large Language Models (LLMs) and related infrastructure used to implement the MINDFORGE framework and conduct the experiments presented in this paper.

All LLM calls, unless explicitly stated otherwise, were executed via API calls to external services. The exception to this is the fine-tuned model, which was ran and trained locally.

The specific models and their corresponding access methods are outlined below:

- **Llama 3.1-8B-Instruct:** This model was accessed through the Lambda Inference API.¹ This API provided a reliable and efficient interface for interacting with the Llama 3.1-8B-Instruct model, enabling seamless integration within the MINDFORGE framework.
- **GPT-4:** Accessed via the OpenAI API. The OpenAI API offered access to the advanced capabilities of the GPT-4 model, crucial for establishing performance baselines and for the expert agent in collaborative settings.
- **Mistral-7B Instruct v0.2, Mixtral-8x7B Instruct and Llama 3.1-70B:** These Mistral models were accessed through the Together AI API.² The Together AI API provided a

¹<https://docs.lambdalabs.com/public-cloud/lambda-inference-api/>

²<https://api.together.ai/models>

platform for utilizing these open-weight models, allowing for a comparative analysis within the Voyager and MINDFORGE frameworks.