
Reimagining ABM with LLM Agents via Shachi

So Kuroki¹ Yingtao Tian¹ Kou Misaki¹ Takashi Ikegami² Takuya Akiba¹ Yujin Tang¹

Abstract

Large language models (LLMs) are increasingly used as agents in agent-based modeling (ABM) to simulate complex social and economic behaviors. However, existing implementations often rely on minimal wrappers around LLMs, lacking structure, modularity, and support for reuse, hindering reproducibility, generalization and exploratory research. We introduce Shachi (鯨), a general and modular agent framework for building and evaluating LLM-driven agents in ABM. Shachi defines a standardized architecture comprising four core components (LLM, tools, memory, and configuration) and supports clear agent-environment and agent-agent interactions. To evaluate its effectiveness, we provide a benchmark suite of tasks spanning three levels of complexity, from single-agent behavior to multi-agent communication. Using Shachi, we reproduce prior LLM-based ABM studies, conduct systematic agent-task cross-evaluations, and demonstrate novel exploratory study results. Shachi enables scalable, reusable, and insightful research at the intersection of LLMs and ABM for social science, and is released as an open-source toolkit. Code: <https://anonymou.s4open.science/r/bench-F6E7>

1. Introduction

Agent-based modeling (ABM) is a widely used methodology for simulating complex systems through the interactions of autonomous agents, and has been applied to fields such as economics, sociology, and political science (Gilbert & Terna, 2000; Gilbert, 2019; Davidsson, 2002). By enabling researchers to explore emergent phenomena and counterfactual scenarios, ABM offers a powerful tool for both theory-building and policy experiments. However, tradi-

tional ABMs often rely on handcrafted rules and heuristics, which can limit realism and interpretability.

Recently, large language models (LLMs) have demonstrated impressive capabilities across a wide range of reasoning, planning, and decision-making tasks (Guo et al., 2025; Kojima et al., 2022; Achiam et al., 2023), leading to a surge of interest in deploying them as agents (Park et al., 2023; Gao et al., 2024; Wang et al., 2024; Anthropic, 2024; Surapaneni et al., 2025). This momentum has naturally extended into ABM (Li et al., 2024; Yang et al., 2024; Wu et al., 2023; Manning et al., 2024), raising hopes that LLMs could alleviate the brittleness and manual effort of traditional agent behavioral designs. Yet, a critical challenge remains: when used out-of-the-box, LLMs often fail to mimic realistic human behavior, undermining the reliability and interpretability of simulation outcomes. As interest in LLM-based agents continues to grow, their design and deployment in ABM demand more structured, careful approaches.

Despite the growing interest in using LLMs for ABM, there is currently no general framework that supports the design, evaluation, and reuse of LLM-based agents for ABM. Most existing works rely on thin wrappers around LLMs, typically using prompt engineering to simulate agent behavior without memory, tool-use, or configurable components. As a result, these ad hoc designs are often tailored to specific tasks, making it difficult to reproduce results, compare methods, transfer agents across domains, or conduct exploratory research. This fragmentation slows progress and raises the barrier for researchers entering the field. What is missing is a unified, modular framework that not only standardizes the internal architecture of LLM agents but also supports rich, extensible behaviors through components like tool-use and persistent memory.

To address these challenges, we propose Shachi (鯨) a general and modular agent framework for LLM-based ABM. Shachi introduces a standardized agent architecture built around four key components: the LLM itself, configs that define agent behaviors and constraints, tools that extend its capabilities, and memory for maintaining context and continuity. While Shachi supports a Gym-style (Brockman et al., 2016) interface for agent-environment interaction, one of its core contributions lies in organizing and modularizing the internal structure of LLM agents. This abstraction allows

¹Sakana AI ²The University of Tokyo. Correspondence to: So Kuroki <sokuroki@sakana.ai>.

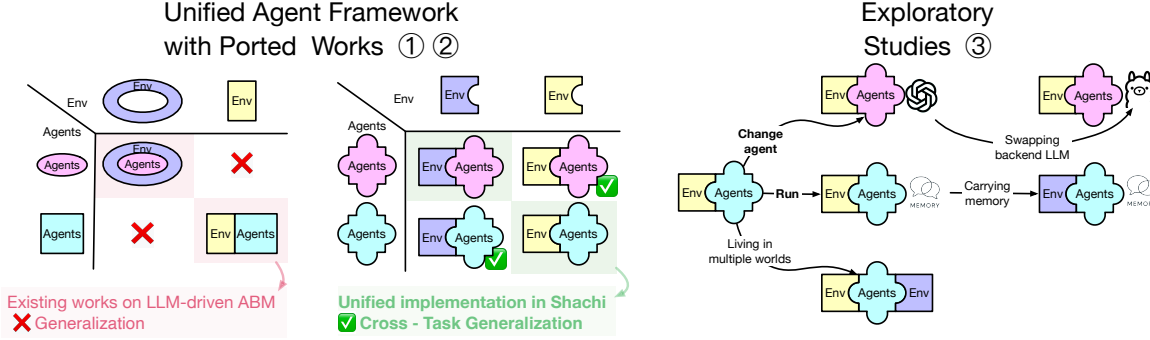


Figure 1: **Unifying LLM-based ABM Research with Shachi.** Shachi is a general agent framework with a benchmark suite that accelerates social science research through LLM-based agents in ABM. Shachi facilitates research in this space by providing ① A unified agent architecture that standardizes core components (LLM, memory, tools, configuration) for modular and reproducible design; ② Cross-task generalization that allows extensive evaluation of different agent designs; and ③ Support for rich exploratory studies such as backend LLM swap, memory transfer and agents living across multiple worlds, enabling complex, layered simulations of human-like behavior.

end-users to focus on the logic and implementation of each component without needing to manage low-level orchestration or integration. As a result, Shachi enables rapid experimentation, encourages the reuse of existing components, and lowers the barrier for building complex, human-like agents in simulation.

Another core contribution of Shachi is the inclusion of ten benchmark tasks with broad coverage, curated to facilitate LLM-based ABM research. These tasks, ported from prior work, are organized into three levels of complexity. Level I consists of single-agent tasks designed to assess how closely an LLM agent can emulate individual human behavior in isolation. Level II progresses to multi-agent scenarios without direct communication, focusing on the emergence of social dynamics and patterns when multiple agents interact indirectly within a shared environment. Finally, Level III introduces inter-agent communication, enabling richer simulations of real-world social and economic systems where coordination, negotiation, and shared context play a critical role. These progressively structured tasks allow researchers to evaluate agent behavior at increasing levels of complexity and realism, and provide a common testing ground for comparing different agent designs.

Figure 1 provides an overview of our proposal. Our key contributions are summarized below:

- **Framework Provision** We introduce Shachi, a unified framework for LLM-based ABM research. To demonstrate its versatility and robustness, we port and reproduce several existing LLM-based ABM studies using diverse agent designs and task structures. This validates Shachi as a reliable foundation for future research and enables broader access to prior work.
- **Agent Design Evaluation** By decoupling agents from tasks via a unified interface, Shachi allows agents devel-

oped for one setting to be applied to others. This enables systematic cross-evaluation of agent designs, revealing that some are more robust and generally effective across tasks, a critical insight for advancing the field.

- **Enabling Novel Inquiries** Shachi supports exploratory experiments that go beyond reproduction or benchmarking. We demonstrate studies involving backend LLM swaps, memory persistence across lives, and cross-environment interactions, highlighting how the framework supports complex, layered investigations into emergent behaviors.

2. Related Works

ABM is a computational approach to simulate interactions among autonomous agents within complex systems, enabling the study of emergent behaviors and social dynamics (Gilbert, 2019). From a computer science perspective, ABM integrates agent-based computing, social sciences, and computer simulation, fostering cross-disciplinary research (Davidsson, 2002). In social sciences, ABM serves as a “third way” of research, complementing argumentation and formalization by enabling the modeling of complex processes and emergent phenomena (Gilbert & Terna, 2000). In economics, ABM has evolved into agent-based computational economics, modeling dynamic economic systems and revealing insights into market behaviors like strategic interactions and collective learning (Tesfatsion, 2006; Tesfatsion & Judd, 2006). Enhancements in agent design through behavioral economics and empirical data integration have made ABM simulations more realistic and applicable to complex social and economic systems (Steinbacher et al., 2021). Please refer to Appendix E.1 for additional related works on conventional ABM. With its wide-ranging applications, ABM remains a promising research area and motivates our work.

Recently, integrating LLMs into ABM has emerged as a promising direction to enhance the realism and adaptability of agents by improving environment perception, human alignment, action generation, and evaluation (Gao et al., 2024; Nisioti et al., 2024). For example, PsychoBench (Huang et al., 2023) evaluates psychological traits; generative agents (Park et al., 2023) simulate interactive social behaviors; and OASIS (Yang et al., 2024), Sotopia (Zhou et al., 2024), and EconAgent (Li et al., 2024) extend these ideas to large-scale simulations and economic and social reasoning. In the agent-based market domain, models like StockAgent (Zhang et al., 2024) and AuctionArena (Chen et al., 2023) test strategic and adaptive decision-making. See Appendix E.2 for more related works. These studies demonstrate that LLM-based agents enhance the fidelity and versatility of ABM across a range of complex scenarios. However, while these studies showcase the potential of integrating LLMs into ABM, they typically lack a standardized and modular framework for designing, evaluating, and reusing LLM-based agents, which this work aims to provide. Concurrent efforts such as MLE-Dojo (Qiang et al., 2025) provide a framework that focuses on iterative machine learning engineering workflows, offering valuable insights into LLM training and evaluation protocols, albeit outside the social simulation and ABM context.

3. Shachi

Shachi provides a standardized and modular framework for developing and evaluating LLM-based agents in ABM. Figure 2 presents an overview of the system architecture, highlighting how agents interact with environments through observations, actions, and communication (see Section 3.1), and how their internal behavior is governed by four key components (see in Section 3.2). Additionally, Shachi supports structured experimentation across three levels of social complexity (see Section 3.3), providing a unified foundation for rigorous and insightful ABM research with LLMs.

3.1. Interaction Design

The design of agent and environment interaction is central to Shachi, covering both the interaction between agents and their environments as well as the communication among agents themselves. All information required for decision-making, such as details on available tools and the expected response format, is passed from the environment to the agent. This design lets agents be implemented in an environment-agnostic manner, enabling those built with Shachi to be evaluated across diverse environments immediately.

Shachi’s environment-agent interface takes inspiration from OpenAI Gym. At each time step, the environment issues observations to each agent, and agents respond with actions. These actions, typically in the form of free text or

structured outputs, are returned to the environment which processes them to advance the simulation state. As with Gym, the simulation proceeds in discrete steps, guided by environment’s STEP() and RESET() methods (see example in Appendix C.1). However, unlike Gym’s fixed action spaces, Shachi supports highly expressive agent outputs and rich observations that include messages, tools, and reward signals. A distinguishing feature of Shachi is that actions refer to decisions whose outcomes are realized in the subsequent steps. In contrast, interactions that require immediate feedback, such as querying a database or asking another agent a question, are not regarded as actions and are typically implemented as tool calls (see example in Appendix C.2) or environment-mediated message passing within the same time step. Taking stock trading as an example, buying/selling a stock is an action since its outcomes (i.e., price movement) are what we care about in the task and are produced in the next step, whereas using tools to check the price of related stocks is not regarded as an action.

A notable design in Shachi is how agent-agent interactions are implemented through the environment. Instead of direct function calls between agents, Shachi encourages indirect interaction via message passing and environment-mediated utilities. For example, an environment may expose a function that allows one agent to send a message to another (this is the case in the OASIS task in our benchmark suite, where an agent talks to others via an environment-specific COMMENT_TO() function). These interactions are then embedded into the observation space and used to simulate realistic social dynamics such as broadcasting, targeted messaging, or asynchronous communication.

To ensure responses are produced in the required format without degrading LLM performance or causing parse errors, we make use of API features such as structured outputs and function calling, along with strategies like two-step parsing (see Appendix C.3 for more details). During interactions, agents make decisions based on both immediate observations and long-term goals or memory. This closely mirrors human decision-making, where individuals act based on both current context and prior experience, and is an essential quality for studying realistic social phenomena in ABM. Their behavior is shaped by four of Shachi’s core components discussed below.

3.2. Core Agent Components

Agents in Shachi are built from four modular components, each of which is inspired by principles in cognitive science and designed for extensibility and experimentation.

LLM serves as the agent’s central behavior generator. Just as human cognition is shaped by language-based reasoning and internal narration (Vygotsky, 2012), LLMs simulate this process by converting observations into natural language or

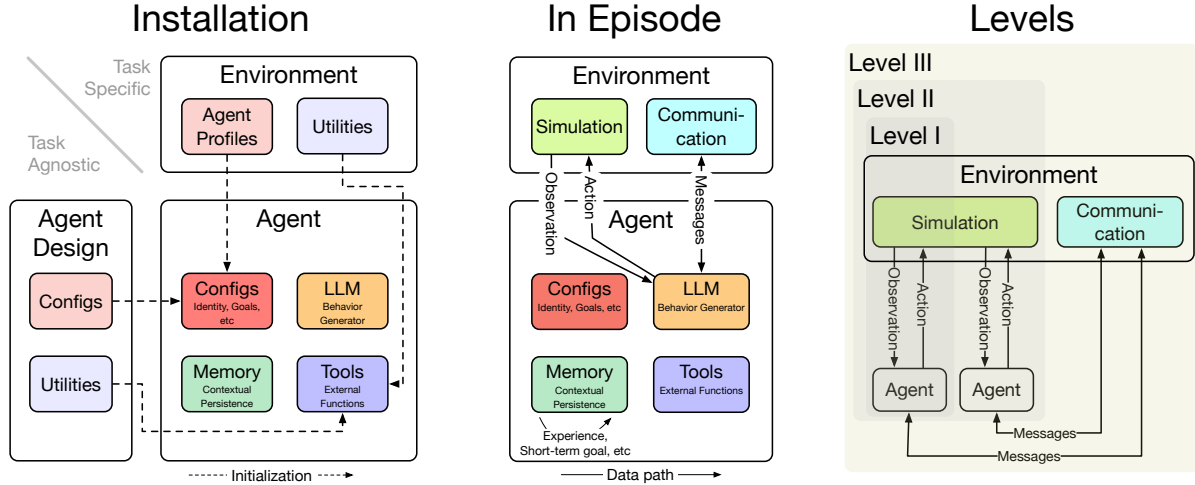


Figure 2: **Shachi Overview.** Each agent is composed of four core components that are organized to simulate realistic, persistent behavior. Agents are installed through task-specific specifications from the environment and task-agnostic ones from agent design (left). The environment mediates both agent-environment interactions and inter-agent communications via structured messages and facilitates simulation. Agents receive immediate feedback via tool interfaces (middle). Shachi supports extensible, reproducible and task-agnostic agent design, enabling rich simulations across diverse ABM tasks in three levels, for single agent, multiple individual agents and multiple interacting agents scenarios respectively (right).

structured responses. In Shachi, the agent turns an observation into a prompt and forwards it to the LLM, which returns an action or message. We support flexible backend substitution, allowing calls to both proprietary APIs and open-source models. Each LLM call is invoked asynchronously, enabling efficient simulation through parallelism.

Configs component defines an agent’s identity, constraints, and behavioral tendencies. Much like psychological traits or roles in human society (McCrae & Costa Jr, 1997), configurations govern how an agent interprets tasks and responds to stimuli. In Shachi, this component can be implemented via prompting strategies or dynamic LoRA (Hu et al., 2022) module loading, which influence LLM decision thresholds or tool access policies. This abstraction allows researchers to simulate diverse agent archetypes or heterogeneous populations with varying roles or incentives.

Memory models contextual persistence, enabling agents to maintain coherence over time. Similar to human working and episodic memory, Shachi’s memory module allows agents to retrieve relevant past interactions and incorporate them into current decisions. This is critical for simulating agents with evolving goals, personalities, or social bonds (Park et al., 2023). Memory implementations in Shachi are abstracted to support various strategies, from simple buffer-based recall to more sophisticated retrieval-augmented or embedding-based approaches. Researchers can modify the memory capacity, retrieval method, or even make memory learnable.

Tools component provides agents with access to external functions or services, reflecting how humans use tools and technologies to augment cognitive tasks (Norman, 2014).

Cognitive science considers tool use a hallmark of intelligent behavior, allowing agents to transcend their intrinsic limitations. In Shachi, tools are provided by environments (task-specific) or from researchers’ toolbox (task-agnostic), and are defined with a name, a schema describing their parameters, and a callable function that executes the desired operation. Agents autonomously decide whether and when to use these tools, along with which arguments to supply, making tool use part of their decision-making process. This component enables researchers to introduce new tools to simulate domain-specific capabilities, context-sensitive environment interactions, or even social interactions.

3.3. Benchmark Suite

To support systematic evaluation of LLM-based agents in ABM settings, we port ten tasks from prior work to form a benchmark suite, and organize them into three levels of interaction complexity. We describe the scope and example tasks for each level below (detailed in Appendices A and B).

Level I tasks involve single-agent behavior in isolation. These environments are designed to assess how closely an individual LLM-based agent can resemble a human when placed in simple, controlled settings. Such tasks offer a clean setup for probing human-likeness in reasoning and behavior, without interference from social context. They are useful for early-stage model validation, fine-tuning experiments, and persona calibration. Shachi includes PsychoBench (Huang et al., 2023), CoMPoS (Cheng et al., 2023), CognitiveBiases (Koo et al., 2024), Emotion-Bench (Huang et al., 2024) and EmergentAnalogies (Webb et al., 2023) at this level.

Level II tasks introduce multiple agents into shared environments, but without direct communication. This level assesses agents’ behaviors when only centralized information is available, and evaluates the agent’s capacity to interpret centralized observations and adapt to emergent dynamics. Such settings are useful for studying social behaviors like implicit coordination or competition, where agents must infer the presence and influence of others indirectly. Tasks include EconAgent (Li et al., 2024), StockAgent (Zhang et al., 2024) and AuctionArena (Chen et al., 2023).

Level III tasks involve rich multi-agent communication. Agents exchange messages (mediated through the environment) and must engage in higher-level social behaviors such as negotiation, teaching, coalition formation, or joint planning. This level tests the interplay between language, memory, and interaction history, and enables the simulation of sophisticated social structures. Evaluation can include both quantitative performance metrics and qualitative assessments of realism, coherence, or fairness. In Shachi, we include OASIS (Yang et al., 2024) and Sotopia (Zhou et al., 2024) as examples at this level.

Together, these benchmark levels offer a structured, extensible platform for studying the performance and behavior of LLM agents across a wide range of ABM settings. These examples serve as starting points for researchers to design their own ABM tasks. Researchers can reuse the provided tasks or contribute new ones following the unified interface.

4. Experiments

To demonstrate Shachi as a useful framework for LLM-based ABM, we identify three key factors:

- **Reproducibility** A credible framework should allow researchers to efficiently reimplement existing LLM-based ABM studies with minimal overhead while preserving the original behaviors and outcomes. We verify this by porting ten representative prior works into Shachi and evaluating how faithfully their qualitative and quantitative results can be reproduced (see Section 4.1).
- **Comparability** By decoupling agents from specific tasks, a unified framework such as Shachi allows controlled comparisons of how different design choices affect agent behavior and task performance. We assess this by testing agents developed for one benchmark in others, revealing which designs generalize effectively (see Section 4.2).
- **Explorativeness** Beyond replication, a useful framework should facilitate new insights, enabling researchers to explore emergent dynamics and generate hypotheses about complex social and economic behavior. We showcase this by analyzing novel agent behaviors and patterns that arise in Level II and III settings (see Section 4.3).

4.1. Reproducing Prior Works

Experimental Setup We selected eight out of the ten tasks for reproduction. These tasks were chosen primarily because they report concrete, quantitative metrics that make it feasible to assess reproduction fidelity in a principled way. All agents were implemented within Shachi using modular combinations of LLM, memory, tools and configuration components, preserving as much of the original behavioral logic as possible. The baselines are variations of the original agents (e.g., removing all the components and making it a thin wrapper of LLM) or agents with randomized behaviors. We refer the reader to Appendix B.1 for detailed settings.

Experimental Results Table 1 shows the reproduction error for all selected tasks, measured by mean absolute error (MAE) between the original and the reproduced results. Across all tasks, Shachi achieves consistently lower error than the baseline, often by a large margin. This indicates that Shachi accurately preserves the original agents’ quantitative outputs, validating its reliability.

While Table 1 provides an aggregated view of reproduction fidelity, we also present “zoomed-in” results in Figure 3 to illustrate how well Shachi captures the dynamics of the original simulations. On the left, we visualize the price trajectories of two stocks in the StockAgent task, showing a close match between the original and Shachi-reproduced trends over time. On the right, we show that Shachi also replicates behavior patterns during the sequential auction of ten items, as shown in the heatmap of priority scores for the remaining items across bidding rounds in AuctionArena. These qualitative visualizations reinforce that Shachi not only matches the original quantitative outputs but also captures the underlying temporal and structural patterns.

4.2. Cross-Task Agent Generalization

Experimental Setup In this setting, we fixed the underlying LLM across all experiments to GPT-4o (gpt-4o-2024-08-06) (Achiam et al., 2023) to ensure consistency, and applied agents originally developed for one task to other tasks. In particular, we study how the presence or absence of configuration, memory, and tool modules affects an agent’s ability to generalize.

We include four agents/tasks in this experiment. Note that the evaluation metrics are task-dependent (see Appendix B), but all scores are normalized relative to each agent’s in-domain performance. Agents differ in composition according to the task they were originally designed for. For example, the agent in EmergentAnalogies consists of an LLM component without any additional structure, while StockAgent includes all components. By evaluating how well each agent performs when deployed in a new task, we aim to understand whether components help or hinder generalization.

Table 1: **Reproduction Results.** We report the mean absolute error (MAE) for the ported tasks. Shachi constantly achieves lower errors when compared with the baselines.

	PsychoBench	CoMPosT	CognitiveBiases	EmotionBench
Baseline	1.96	0.23	0.24	13.82
Shachi (Ours)	0.80	0.06	0.04	3.37

	EmergentAnalogies	StockAgent	AuctionArena	Sotopia
Baseline	0.64	9.07	10.49	3.17
Shachi (Ours)	0.05	2.63	2.22	0.95

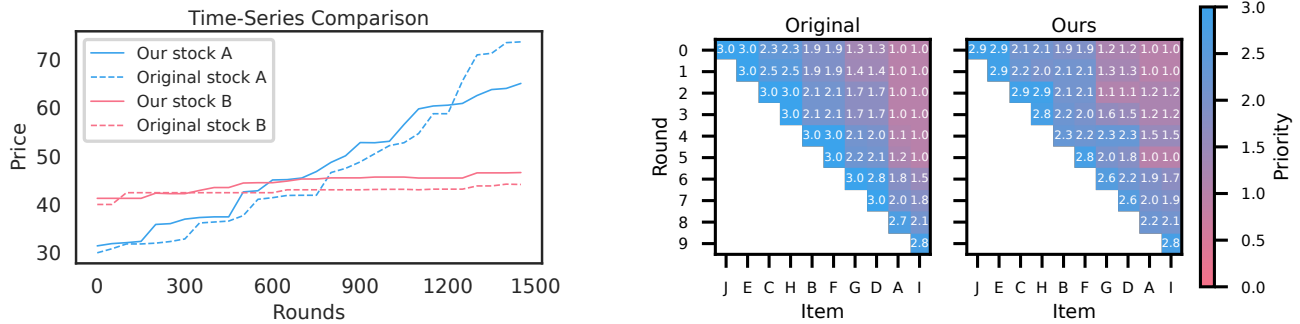


Figure 3: **Reproducing System Dynamics.** Left: Evolution of the two stock prices in StockAgent, comparing the original study with our Shachi reproduction; Right: Reported versus reproduced heatmaps of mean priority scores and their changes before each bidding round in AuctionArena (descending). Together, these results illustrate how well Shachi captures the dynamics of the original simulations.

Experimental Results We report the normalized performance of each agent-task pairing in Table 2. Each column corresponds to a target task, and each row to an agent, with its components listed in the parentheses. Scores are normalized such that the in-domain performance (diagonal entries) is always one, allowing for easier interpretation of relative effectiveness across tasks.

Two immediate observations emerge from the table: (1) The EmergentAnalogies task requires only a minimal agent consisting solely of an LLM without additional components. As a result, all agents’ performance is similar on this task. This suggests that when the target task is simple and does not require auxiliary capabilities like tool use or configuration adaptation, minimalist agents can generalize sufficiently well. (2) Sotopia represents a complex multi-agent communication setting that originally uses memory. We expected the performance from an EmergentAnalogies to be different from the other three agents on this task due to the lack of a memory module, but to our surprise, all transferred agents achieve values close to 1. One possible explanation is that memory was not effectively leveraged even in the original setup, or that the LLM’s intrinsic context window suffices for short-term coherence.

More nuanced patterns appear in the mid-table entries. For example, the agent from StockAgent includes all four components and transfers reasonably well to all other tasks (2nd row). On the other hand, the agent from AuctionArena demonstrated similar performance on tasks except for Stock-

Agent (3rd row), which requires tool usage, a component that the AuctionArena agent does not have. These results suggest that the presence or absence of the tool component leads to the observed behavioral differences.

4.3. Exploratory Studies

Shachi’s modular structure enables exploratory investigations that go beyond reproduction and transfer. We highlight three studies that analyze how different backend LLMs affect agent behavior, examine whether memory carried across simulations can alter behavior or characteristics, and demonstrate how complexity can be built gradually in Shachi by making agents live in multiple worlds.

4.3.1. IMPACT OF BACKEND LLMs

Experimental Setup For EconAgent, we replaced the single model in the original study with widely used models. All other settings remain unchanged: an economy of 100 simulated agents evolves over 240 months, and we track both standard macro-indicator trajectories and the emergence of two canonical macroeconomic regularities: the Phillips Curve (Phelps, 1967) and Okun’s Law (Okun, 1963).

Experimental Results Figure 4 visualizes how the choice of backend LLM affects macroeconomic patterns and indicators. All LLM-based agents collectively show behaviors in accordance to both Phillips Curve and Okun’s Law. Nonetheless, the curves differ in detail: intercept shifts in

Table 2: **Cross-Task Agent Generalization.** Scores in each column are normalized against the one on the diagonal. Agents with all the components (i.e., StockAgent) maintain stable performance when transferred to other tasks.

	EmergentAnalogies	StockAgent	AuctionArena	Sotopia
EmergentAnalogies	1.00	1.08	0.62	1.01
StockAgent (config, mem, tool)	1.01	1.00	0.99	1.00
AuctionArena (config, mem)	1.00	0.93	1.00	0.99
Sotopia (mem)	1.00	0.93	0.92	1.00

the Phillips Curve point to varying baseline unemployment rates, while slope changes in Okun’s Law indicate differing GDP-unemployment trade-offs. Moreover, although most LLM backends produce similar variations of indicators, GPT-4.1 Nano yields consistently higher unemployment, and GPT-4.1 delivers markedly stronger GDP growth. Our Shachi framework’s modular separation of concerns allows easy backend change with a single configuration line, unlike ad-hoc setups, where replacing the LLM typically means re-implementing or realigning several parts of the agent pipeline. This streamlined flexibility enables cleaner comparisons and faster iteration, like what we find here.

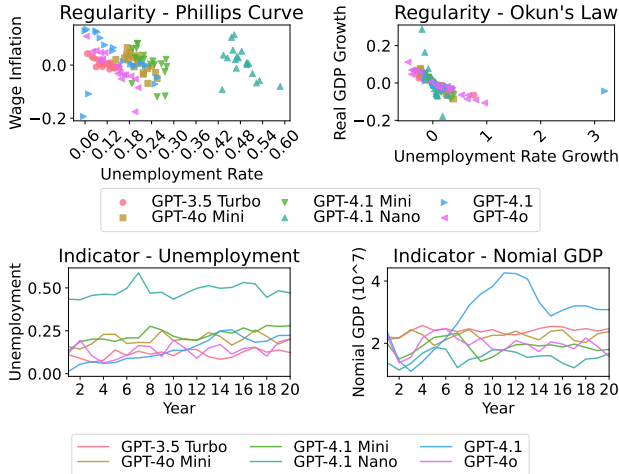


Figure 4: **Impact of Different LLMs.** Macroeconomic regularities and indicators in EconAgent. All LLM-based agents collectively show correct behaviors in regularities and similar indicators, yet emerging behaviours differ in details, showing characteristics of different LLMs.

4.3.2. CARRYING MEMORY TO THE NEXT LIFE

Experimental Setup In these studies, we used GPT-4o-mini as our LLMs and transferred agents from OASIS and EconAgent tasks without clearing their memories (i.e., their stream memories contain the observations and experiences from these tasks) to the CognitiveBiases task. The experiments were conducted multiple times to gauge statistical significance in our findings.

Experimental Results Figure 5 shows that carry-over memories alter performance in the CognitiveBiases task. Notably, memories from OASIS and EconAgent lead to dis-

tinct shifts in cognitive biases. Among the largest changes, in OASIS, short-term reactions and community pressure could amplify *Hyperbolic Discounting* and *In-Group Bias*. Agents learned to heavily prioritize immediate feedback (leading to higher hyperbolic discounting) and reinforce group identities (raising in-group bias). In EconAgent, repeated interactions and asset ownership may heighten the *Endowment Effect* while diminishing *Loss Aversion* and *Survivorship Bias*. Because agents regularly gain experience with actual gains and losses, they become more attached to assets they already hold (leading to a stronger endowment effect) and become less sensitive to losses and optimistic success rates (lowering loss aversion and survivorship bias).

4.3.3. LIVING IN MULTIPLE WORLDS

Experimental Setup We introduce a set of shared agents that operate across two distinct environments: StockAgent and OASIS. Unlike typical settings where agents act in only one environment, agents here cycle between both. In each cycle, agents first observe the StockAgent environment, make trading decisions, and act. Then, the group moves to OASIS, where agents observe the social media, respond, and act. This loop repeats, with each agent carrying internal states across environments, allowing knowledge or strategies learned in one context to influence behavior in the other. StockAgent settings follow those in Section 4.1, with stock A representing a 10-year chemical stock and stock B representing a 3-year tech stock. The OASIS environment is described in Appendix B.1.10, where the main topic is about Amazon’s newly established physical stores.

Experimental Results We found that allowing agents to participate in multiple environments lead to emergent behaviors that reflect cross-domain influence. Figure 6 shows that, with OASIS present, stock prices rise less than in the StockAgent-only setting. This is surprising to us, as we expected that introducing the Amazon topic would cause agents to buy more of the tech-related stock B, leading to a surge in its price. On the other hand, Table 3 gives more detailed information under the hood: (1) Introducing social media increased market activity, as is evidenced by the increased volume in both stocks; (2) The introduced Amazon topic caused agents to show more willingness to buy the tech stock and to become reluctant to sell, which is supported by the increase in #Buys and the decrease in #Sells for stock B; (3) In contrast, this topic increased both the willingness to

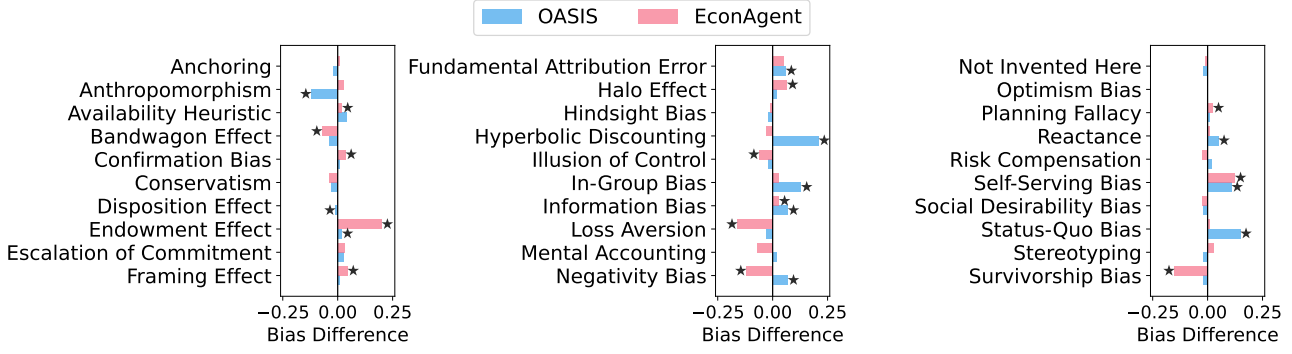


Figure 5: **Memory-transfer-induced Differences in the CognitiveBiases Task.** For each bias, the difference is calculated as the score obtained with carry-over memory minus the score obtained with a fresh memory. Statistically significant differences are indicated with star-shaped markers.

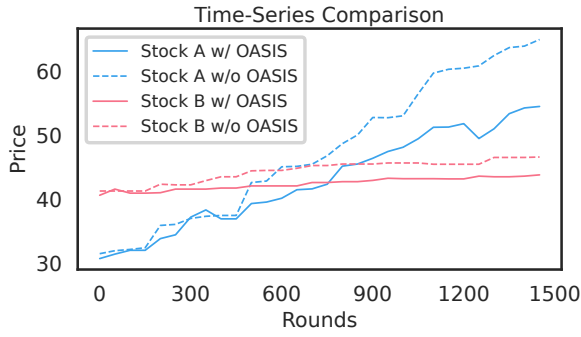


Figure 6: **Comparison of Price Movements.**

Table 3: **Change in Key Indicators.** Volume refers to the number of matched orders, while #Buys/#Sells are the numbers of submitted buy/sell orders. Changes are computed as the difference between treatment (w/ OASIS) and control (w/o OASIS) conditions.

Indicator	% Δ (Stock A)	% Δ (Stock B)
Volume	+10.0%	+20.0%
#Buys	+8.4%	+6.1%
#Sells	+6.6%	-8.5%

buy and sell stock A. This information is more in line with our intuition than with the price movements.

In terms of agent behavioral changes in OASIS, we observe that some agents began tweeting about Amazon stock, blending financial observations with social commentary. Other agents responded with comments, echoing and endorsing the post, suggesting that economic actions in one environment can organically propagate into social discourse in another. Please refer to Section B.2 for the posts and comments.

Our results (averaged over three independent trials) in this experiment reveal that while agent-level behaviors align with intuitive financial logic, system-level outcomes do not necessarily follow, highlighting the importance of empirical simulation in social science research. At the same time, our exploratory study demonstrates Shachi’s capacity for rich, layered simulations by supporting agents that live across multiple interconnected worlds.

5. Conclusion

Summary In this paper, we introduced Shachi, a general and modular framework for building LLM-based agents in ABM. Shachi standardizes agent architecture into four components (LLM, tools, memory, configuration) and sup-

ports structured interactions with the environment and other agents. Through experiments, we demonstrated that Shachi enables reproducible implementation of prior studies, facilitates agent design evaluation via cross-task transfer, and supports exploratory studies that uncover new insights into social and economic behaviors.

Limitation While Shachi defines four core components that structure the agent’s behavior, they may fall short for autonomous agents in complex settings. Currently, agent behavior is shaped mainly through prompting, which can guide short-term actions but lacks persistence or internal consistency. As a result, important elements like motivational systems or long-term value alignment are not explicitly modeled.

Future Works Introducing persistent internal state, such as a learnable value system or motivational model, may be an effective way to address the limitation. This would allow agents to develop and adapt preferences or goals over time, beyond what can be encoded through prompting alone. In addition, expanding Shachi to support multi-modal environments and interactions is also an interesting direction for future works, enabling more realistic and immersive simulations that reflect the richness of real-world social behavior.

References

- Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- Anderson, P. W. *The economy as an evolving complex system*. CRC Press, 2018.
- Anthropic. Introducing the model context protocol. <https://www.anthropic.com/news/model-context-protocol>, 2024. Accessed: 2025-05-09.
- Arthur, W. B. Inductive reasoning and bounded rationality. *The American economic review*, 84(2):406–411, 1994.
- Axelrod, R. The dissemination of culture: A model with local convergence and global polarization. *Journal of conflict resolution*, 41(2):203–226, 1997.
- Axelrod, R. and Hamilton, W. D. The evolution of cooperation. *science*, 211(4489):1390–1396, 1981.
- Axtell, R. L. Zipf distribution of us firm sizes. *science*, 293(5536):1818–1820, 2001.
- Bonabeau, E. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the national academy of sciences*, 99(suppl_3):7280–7287, 2002.
- Bonnotte, N. *Unidimensional and evolution methods for optimal transportation*. PhD thesis, Université Paris Sud-Paris XI; Scuola normale superiore (Pise, Italie), 2013.
- Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.
- Chen, J., Yuan, S., Ye, R., Majumder, B. P., and Richardson, K. Put your money where your mouth is: Evaluating strategic planning and execution of llm agents in an auction arena. *arXiv preprint arXiv:2310.05746*, 2023.
- Cheng, M., Piccardi, T., and Yang, D. Compost: Characterizing and evaluating caricature in llm simulations. *Annual Meeting of the Association for Computational Linguistics*, 2023.
- Cohen, M. D., March, J. G., and Olsen, J. P. A garbage can model of organizational choice. *Administrative science quarterly*, pp. 1–25, 1972.
- Davidsson, P. Agent based social simulation: A computer science view. *Journal of artificial societies and social simulation*, 5(1), 2002.
- Epstein, J. M. and Axtell, R. *Growing artificial societies: social science from the bottom up*. Brookings Institution Press, 1996.
- Farmer, J. D. and Foley, D. The economy needs agent-based modelling. *Nature*, 460(7256):685–686, 2009.
- Gao, C., Lan, X., Li, N., Yuan, Y., Ding, J., Zhou, Z., Xu, F., and Li, Y. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1–24, 2024.
- Gilbert, N. *Agent-based models*. Sage Publications, 2019.
- Gilbert, N. and Terna, P. How to build and use agent-based models in social science. *Mind & Society*, 1:57–72, 2000.
- Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.
- Huang, J., Lam, M. H., Li, E. J., Ren, S., Wang, W., Jiao, W., Tu, Z., and Lyu, M. R. Apathetic or empathetic? evaluating LLMs’ emotional alignments with humans. In *Advances in Neural Information Processing Systems 37*, 2024.
- Huang, J.-t., Wang, W., Li, E. J., Lam, M. H., Ren, S., Yuan, Y., Jiao, W., Tu, Z., and Lyu, M. On the humanity of conversational ai: Evaluating the psychological portrayal of llms. In *The Twelfth International Conference on Learning Representations*, 2023.
- Kirman, A. Ants, rationality, and recruitment. *The Quarterly Journal of Economics*, 108(1):137–156, 1993.
- Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213, 2022.
- Koo, R., Lee, M., Raheja, V., Park, J. I., Kim, Z. M., and Kang, D. Benchmarking cognitive biases in large language models as evaluators. *Annual Meeting of the Association for Computational Linguistics*, 2024.
- Li, N., Gao, C., Li, M., Li, Y., and Liao, Q. Econagent: large language model-empowered agents for simulating macroeconomic activities. *Annual Meeting of the Association for Computational Linguistics*, 2024.

- Manning, B. S., Zhu, K., and Horton, J. J. Automated social science: Language models as scientist and subjects. Technical report, National Bureau of Economic Research, 2024.
- McCrae, R. R. and Costa Jr, P. T. Personality trait structure as a human universal. *American psychologist*, 52(5):509, 1997.
- Nisioti, E., Glanois, C., Najarro, E., Dai, A., Meyerson, E., Pedersen, J. W., Teodorescu, L., Hayes, C. F., Sudhakaran, S., and Risi, S. From text to life: On the reciprocal relationship between artificial life and large language models. In *Artificial Life Conference Proceedings 36*, volume 2024, pp. 39, 2024.
- Norman, D. *Things that make us smart: Defending human attributes in the age of the machine*. Diversion Books, 2014.
- Okun, A. M. *Potential GNP: its measurement and significance*. Cowles Foundation for Research in Economics at Yale University, 1963.
- Park, J. S., O’Brien, J., Cai, C. J., Morris, M. R., Liang, P., and Bernstein, M. S. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.
- Phelps, E. S. Phillips curves, expectations of inflation and optimal unemployment over time. *Economica*, pp. 254–281, 1967.
- Qiang, R., Zhuang, Y., Li, Y., K, D. S. V., Zhang, R., Li, C., Wong, I. S.-H., Yang, S., Liang, P., Zhang, C., and Dai, B. Mle-doj: Interactive environments for empowering llm agents in machine learning engineering, 2025. URL <https://arxiv.org/abs/2505.07782>.
- Sakoda, J. M. The checkerboard model of social interaction. *The Journal of Mathematical Sociology*, 1(1):119–132, 1971.
- Schelling, T. C. Dynamic models of segregation. *Journal of mathematical sociology*, 1(2):143–186, 1971.
- Steinbacher, M., Raddant, M., Karimi, F., Camacho Cuenca, E., Alfarano, S., Iori, G., and Lux, T. Advances in the agent-based modeling of economic and social behavior. *SN Business & Economics*, 1(7):99, 2021.
- Surapaneni, R., Jha, M., Vakoc, M., and Segal, T. Announcing the agent2agent protocol (a2a). <https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interopability/>, April 2025. Accessed: 2025-05-09.
- Takata, R., Masumori, A., and Ikegami, T. Spontaneous emergence of agent individuality through social interactions in large language model-based communities. *Entropy*, 26(12):1092, 2024.
- Tesfatsion, L. Agent-based computational economics: A constructive approach to economic theory. *Handbook of computational economics*, 2:831–880, 2006.
- Tesfatsion, L. and Judd, K. L. *Handbook of computational economics: agent-based computational economics*, volume 2. Elsevier, 2006.
- Vygotsky, L. S. *Thought and language*, volume 29. MIT press, 2012.
- Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y., et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, 2024.
- Webb, T., Holyoak, K. J., and Lu, H. Emergent analogical reasoning in large language models. *Nature Human Behaviour*, 7(9):1526–1541, 2023.
- Wu, P. Y., Nagler, J., Tucker, J. A., and Messing, S. Large language models can be used to estimate the latent positions of politicians. *arXiv preprint arXiv:2303.12057*, 2023.
- Yang, Z., Zhang, Z., Zheng, Z., Jiang, Y., Gan, Z., Wang, Z., Ling, Z., Chen, J., Ma, M., Dong, B., et al. Oasis: Open agents social interaction simulations on one million agents. *arXiv preprint arXiv:2411.11581*, 2024.
- Zhang, C., Liu, X., Zhang, Z., Jin, M., Li, L., Wang, Z., Hua, W., Shu, D., Zhu, S., Jin, X., et al. When ai meets finance (stockagent): Large language model-based stock trading in simulated real-world environments. *arXiv preprint arXiv:2407.18957*, 2024.
- Zheng, S., Trott, A., Srinivasa, S., Parkes, D. C., and Socher, R. The ai economist: Taxation policy design via two-level deep multiagent reinforcement learning. *Science advances*, 8(18):eabk2607, 2022.
- Zhou, X., Zhu, H., Mathur, L., Zhang, R., Yu, H., Qi, Z., Morency, L.-P., Bisk, Y., Fried, D., Neubig, G., et al. Sotopia: Interactive evaluation for social intelligence in language agents. *International Conference on Learning Representations*, 2024.

A. Tasks in Shachi

Table 4: **Overview of Tasks and Their Levels.** Detailed task information is provided in Section B.

#	Lvl	Task	Description
1	I	PsychoBench (Huang et al., 2023)	Evaluates psychological traits of LLMs via 13 psychometric scales across personality, interpersonal, motivational, and emotional domains
2	I	CoMPosT (Cheng et al., 2023)	Measures LLM simulations’ susceptibility to caricature through several dimensions
3	I	CognitiveBiases (Koo et al., 2024)	Evaluates 30 classic cognitive biases with paired control–treatment prompts
4	I	EmotionBench (Huang et al., 2024)	Measures shifts in eight core emotions triggered by situational prompts
5	I	EmergentAnalogies (Webb et al., 2023)	Probes zero-shot analogical reasoning across matrix, string, verbal, and story tasks
6	II	EconAgent (Li et al., 2024)	LLM-powered multi-agent system for macroeconomic simulation with human-like behaviors
7	II	StockAgent (Zhang et al., 2024)	LLM-based multi-agent system that simulates real-world stock trading under dynamic market conditions.
8	II	AuctionArena (Chen et al., 2023)	Evaluates strategic planning and adaptive reasoning of LLM agents in simulated dynamic auctions
9	III	OASIS (Yang et al., 2024)	Large-scale multi-agent simulation benchmark designed for studying social media phenomena
10	III	Sotopia (Zhou et al., 2024)	Open-ended role-play environment to simulate complex social interactions and measure agents’ social intelligence

B. Extra Experimental Setups and Results

B.1. Detailed Task Implementation

B.1.1. PSYCHOBENCH (LEVEL I)

Description PsychoBench (Huang et al., 2023) evaluates the psychological portrayal of LLMs, drawing from psychometric research to examine their human-like psychological traits.

Method It systematically measures thirteen psychological dimensions categorized into personality traits (e.g., Big Five Inventory, Dark Triad), interpersonal relationships (e.g., Bem’s Sex Role Inventory), motivational tendencies (e.g., General Self-Efficacy), and emotional abilities (e.g., Emotional Intelligence Scale). The methodology involves administering psychometric scales directly via prompts. Crucial experimental parameters include detailed instructions for Likert-scale responses, randomized question order to ensure robustness, and strict control of model inference temperature (set to zero or near-zero).

Experimental Settings For the reproduction study reported in Table 1, we follow the original setup and use Llama-2-13b-chat-hf as the LLM component without a memory component. The task includes neither config nor tool components. We fix the LLM temperature to 0, test 10 random seeds for question ordering, and average the results. We compute the MAE between the scores obtained with Shachi (Ours) and those presented in the original paper across all psychological subscales. We add a naive baseline that answers each prompt by randomly selecting from the available choices. It takes a few minutes for the LLM to complete all psychometric scales.

B.1.2. COMPOST (LEVEL I)

Description CoMPosT (Cheng et al., 2023) investigates how susceptible large language models (LLMs) are to caricature.

Method To quantify this effect, the framework decomposes caricature into four orthogonal dimensions—*context*, *model*, *persona*, and *topic*—which specify the simulated scenario, the LLM configuration, the target opinion, and the domain of discourse, respectively. Two metrics are introduced: the *individuation score*, which tests whether the simulated persona is distinguishable from the default persona, and the *exaggeration score*, which measures the degree to which the simulation amplifies persona–topic features.

Experimental Settings In our reproduction study (Table 1), we use GPT-3.5-turbo (`gpt-3.5-turbo-0125`) as the LLM component without a memory component. The task includes neither config nor tool components. We compare the scores produced by our implementation (and a baseline) with those reported in the original paper. Although we replicate the experimental setup as faithfully as possible, minor differences arise owing to updates in the underlying LLMs. Each method outputs a distribution of scores. We assess similarity between two such distributions via one-dimensional optimal transport with the absolute difference as the cost function. In this setting, the transport cost simplifies to MAE between the paired, sorted samples of the distributions, following Bonnotte (2013). The evaluation takes approximately 20 minutes.

B.1.3. COGNITIVEBIASES (LEVEL I)

Description CognitiveBiases (Koo et al., 2024) evaluates how LLMs exhibit 30 well-known cognitive biases, motivated by the increasing use of LLMs in high-stakes decision-making.

Method It specifically measures biases such as anchoring, framing, and 28 others commonly identified in psychology and behavioral economics. The core methodology employs a systematic framework that generates and administers 30,000 bias-specific test cases across 200 distinct decision-making scenarios, comparing model responses under control vs. treatment conditions. Crucial parameters include the explicit control/treatment designs for each bias, two standardized answer scales (7-point Likert or 11-point percentage), and reversed option orders to account for position bias, ensuring reproducibility and comprehensive coverage.

Experimental Settings For the reproduction study in Table 1, we follow the original setup and use GPT-4o-mini (`gpt-4o-mini-2024-07-18`) as the LLM component without a memory component. The task includes neither config nor tool components. The LLM temperature is fixed at 0. We run the evaluation under three random seeds for question ordering and report the average. We measure MAE between the bias scores obtained with Shachi (ours) and those in the original paper across 30 biases. We include a naive baseline that responds to each prompt by randomly sampling from the available choices. Evaluating all 30 biases requires roughly one hour.

B.1.4. EMOTIONBENCH (LEVEL I)

Description EmotionBench (Huang et al., 2024) evaluates how LLMs respond emotionally to various real-life situations, drawing from emotion appraisal theory to examine their alignment with human-like emotional reactions.

Method It measures eight key positive and negative emotions (anger, anxiety, depression, frustration, jealousy, guilt, fear, embarrassment) and tracks how situational contexts raise or lower these emotions compared to a default baseline. It uses self-report scales (e.g., PANAS), first measuring a model’s default emotional state, then presenting situational prompts, and finally re-measuring changes in emotional scores.

Experimental Settings For the reproduction study reported in Table 1, we follow the original setup and use GPT-3.5-turbo (`gpt-3.5-turbo-0125`) as the LLM component without a memory component. The task includes neither config nor tool components. We compute MAE between the eight key positive and negative emotion scores obtained with Shachi (Ours) and those reproduced with their original code on the PANAS scale. We add a naive baseline that answers each prompt by randomly selecting from the available choices. Evaluating the LLM’s affective state takes roughly one minute.

B.1.5. EMERGENTANALOGIES (LEVEL I)

Description EmergentAnalogies (Webb et al., 2023) evaluates zero-shot analogical reasoning in LLMs, highlighting analogy’s key role in fluid intelligence.

Method The benchmark tests a range of domains for abstract pattern induction and relational reasoning, featuring four core tasks—matrix reasoning, letter-string analogies, four-term verbal analogies, and story analogies. We specifically target free-response accuracy on the matrix reasoning.

Experimental Settings For our reproduction study in Table 1, we use GPT-4 (gpt-4-0613) as the LLM component without a memory component. The task includes neither config nor tool components. We set the LLM temperature to 0 and randomly sampled problems from each category. We use three different seeds for sampling and averaged the results. We compute MAE between the category-wise averages obtained with Shachi (Ours) and those reproduced with their source code. As a naive baseline, we include a model that simply generates a random matrix for each prompt. For our cross-task agent study in Table 2, we utilize GPT-4o (gpt-4o-2024-08-06). We compute an overall average across all categories. It takes approximately one minute to assess matrix reasoning.

B.1.6. ECONAGENT (LEVEL II)

Description Econagent (Li et al., 2024) is a LLM-powered multi-agent system for macroeconomic simulation with human-like behaviors.

Method Building on the virtual economic framework of (Zheng et al., 2022), it employs an economic environment where each agents are placed into a shared, quasi-realistic market with an endowment of specific skills and wealth. Agents decide how much to work and consume, and their decisions collectively produce macroeconomic dynamics. A rule-based environment acts as both a central government (collecting taxes) and a central bank (adjusting interest rates), forming a macroeconomic loop. The original work shows that LLMs-powered agents make realistic decisions individually and, collectively, produce sounding macro-level dynamics.

Experimental Settings In our experiments in Section 4.3.1, we use several models as the LLM component without a memory component. The task includes neither config nor tool components. We follow the original paper of simulating 100 agents for 240 months, making 20 annual tax-and-monetary economic cycles. The simulation takes approximately one hour to execute with our Shachi-based implementation. Unlike the original paper, which employs a single LLM model, our replication evaluates a widely used model suite. For the carrying memory experiment in Section 4.3.2, we add a buffer memory to record each agent’s behaviors and transfer an agent to the CognitiveBiases task.

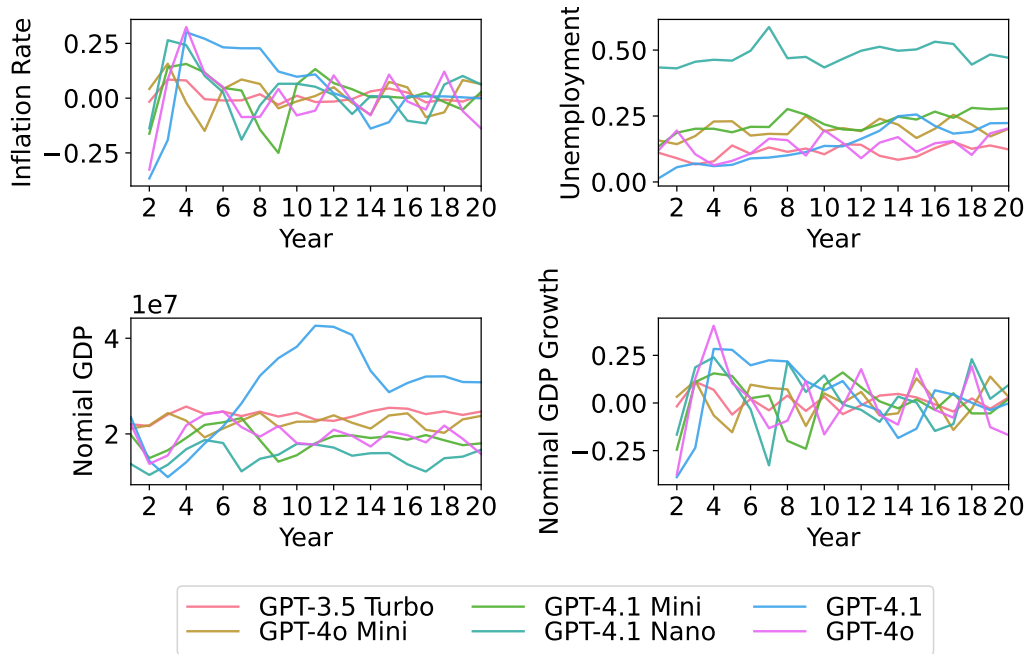


Figure 7: **All Marco-economic Indicators.** These are extra results accompanying those in Section 4.3.1.

Extra Results In Figure 7 we share all macro-economic indicators. All LLMs show a change in indicators, yet emerging behaviours differ in detail, showing characteristics of different LLMs.

B.1.7. STOCKAGENT (LEVEL II)

Description StockAgent (Zhang et al., 2024) is a large language model-based multi-agent system that simulates real-world stock trading under dynamic market conditions in a single line.

Method Specifically, it runs event-driven simulations where LLM-driven agents sequentially make loans, buy, sell, predict, and post and check forum decisions, while the market data and stock prices evolve daily. The framework models two distinct stocks: Stock A, a 10-year chemical stock, and Stock B, a 3-year tech stock, dynamically simulating their price fluctuations. Notable parameters include initial agent capital allocations, loan-to-value ratios, interest rates, and real-world-like events (e.g., financial reports).

Experimental Settings In the reproduction study reported in Table 1 and Figure 3, we follow the original setup and use GPT-3.5-turbo (gpt-3.5-turbo-0125) as the LLM component with buffer memory component (length 3). The task contains both config and tool components. Our Shachi implementation exposes a forum API as the tool: each agent autonomously decides whether to read comments posted the previous day about the stock market. For config, the environment assigns each agent one of four investment styles—*Conservative*, *Aggressive*, *Balanced*, or *Growth-Oriented*. We fixed random seeds and event sequences to match the original study. The simulation runs for 10 days with three sessions per day, involving 50 agents and 1,500 rounds. We average session-level price changes over three seeds and compute MAE between the session-by-session price dynamics of stocks A and B produced by Shachi (ours) and those reproduced with the authors’ source code. As a baseline, we remove the tool and memory modules from the agents to isolate their impact on performance. For our cross-task agent study in Table 2, we deploy 50 agents—25 powered by GPT-4o (gpt-4o-2024-08-06) and 25 by GPT-3.5-turbo (gpt-3.5-turbo-0125). To quantify volatility, we calculate the price change rate for Stocks A and B between the first and final sessions and average the two rates. Simulating 1,500 rounds required several hours.

B.1.8. AUCTIONARENA (LEVEL II)

Description AuctionArena (Chen et al., 2023) evaluates the strategic planning and execution capabilities of LLM agents within a dynamic auction environment, motivated by the need for realistic benchmarks of sequential decision-making in competitive scenarios.

Method The environment specifically assesses skills such as resource allocation, risk management, and adaptive strategic reasoning. The methodology employs a simulation of open ascending-price auctions where agents act as bidders, making decisions based on the Belief-Desire-Intention (BDI) framework. Crucial parameters include item valuation (distinguishing between cheap and expensive items), intentional overestimation of item value to simulate “winner’s curse”, and explicit prioritization strategies that agents dynamically adjust after each round.

Experimental Settings For the reproduction study reported in Table 1, we follow “Standard Competition” setting, where the evaluated agent (GPT-4-turbo, gpt-4-1106-preview) competes directly against GPT-3.5-turbo (gpt-3.5-turbo-1106) and GPT-4-turbo (gpt-4-1106-preview) agents. All agents use a chat-history memory (window 20, token 10,000). The environment supplies a config component but no tools. As for the config, each agent is assigned one of two strategies: profit-first — maximize final profit, or item-first — win the most items. We fix the temperature at 0, randomly assign the item order, and use the TrueSkill system to quantify agent performance across 10 auctions with different item sequences. As a baseline, we repeat this comparison while removing the agent’s memory module to isolate its impact on performance. As for Figure 3, we use the result for GPT-4 (gpt-4-0613) agent in the competition against GPT-3.5-turbo (gpt-3.5-turbo-0125) and GPT-4 (gpt-4-0613), with items ordered by descending price. For our cross-task agent study in Table 2, we use a TrueSkill score of GPT-4o (gpt-4o-2024-08-06) in “Standard Competition” setting. A three-agent, ten-item auction with a \$20,000 budget takes about 20 minutes to run.

B.1.9. SOTOPIA (LEVEL III)

Description Sotopia (Zhou et al., 2024) Sotopia introduces an open-ended role-play environment with a multidimensional evaluation framework to simulate complex social interactions and systematically measure LLM agents’ social intelligence.

Method In the original Sotopia implementation, at every turn, it concatenates the entire dialogue history from all agents into a single prompt. In Shachi, by contrast, memory management is an agent-side responsibility, so the environment supplies only the most recent message. The evaluation result consists of seven metrics (SOC, SEC, FIN, REL, KNO, GOAL, and BEL).

Experimental Settings For the Table 1 experiments, we follow the original setup and use GPT-4 (gpt-4-0613) as the LLM component with a buffer memory component (all history with a 16,000 token limit). The task includes neither config nor tool components. To faithfully reproduce Sotopia inside Shachi, our reproduced agent (the agent reported as “Shachi (Ours)” in Table 1 and the “Sotopia” line in Table 2) restructures the full conversation history into one prompt in the same format. We set the LLM temperature to 1. We consider the MAE between the result reported in the original paper and our experiments. For our cross-task agent study in Table 2, we utilize GPT-4o (gpt-4o-2024-08-06). We apply min-max normalization using the maximum and minimum values defined for each metric and then calculate the overall average across these metrics. The evaluation takes approximately 20 minutes.

B.1.10. OASIS (LEVEL III)

Description OASIS (Yang et al., 2024) is a large-scale multi-agent simulation benchmark for studying how up to one million LLM-based agents interact on social media platforms, focusing on information propagation, group polarization, and herd effects.

Method OASIS simulates large-scale social media environments by combining an environment server, a recommendation system, and a time engine. Each user is modeled as an LLM-based agent with a 21-type action space (e.g., posting, commenting, following), whose behavior and memory evolve in real time. By supporting up to one million agents, OASIS facilitates the study of complex emergent phenomena, such as information spreading, group polarization, and herd effects, in both X and Reddit-like settings. In our experiment, we utilize an X-like setting.

Experimental Settings In our experiments, OASIS is employed in both Sections 4.3.2 and 4.3.3 under the same setup, following one of the original settings. Specifically, we use one influential agent who posts about Amazon (“report: amazon plans to open its first physical store in new york URL”), and its followers respond, with limited connectivity among the followers themselves. We use several models as the LLM component based on experiments with a chat-history memory (window 5, token limit 100,000). The environment supplies a config component but no tools. As for the config, each agent is assigned a distinct profile, such as “@ohiostate alumni. High tech marketer, salon entrepreneur, & web design enthusiast. Fashion & food are my passions. Views are my own.” In Section 4.3.2, we use GPT-4o-mini (gpt-4o-mini-2024-07-18), while Section 4.3.3 employs GPT-3.5-turbo (gpt-3.5-turbo-0125) and GPT-4o (gpt-4o-2024-08-06). Ten iterations with 111 agents complete in a few minutes.

B.2. Detailed Results from Living in Multiple Worlds

The following box shows the agent logs discussed in Section 4.3.3, where the agents’ posts and comments are labeled in blue and red. These agents inhabit two distinct worlds.

Detailed behaviors in OASIS when the agents live in both StockAgent and OASIS

- **create_post:** user_id=14, current_time=3, action=create_post, info={'content': "Agent plan: I find it quite fascinating as a stock trader to see Amazon making a move into physical retail. This could potentially impact their stock, especially if they successfully blend the convenience of online shopping with the tactile experience of a physical store. It's also worth noting how this decision might influence other online retailers to consider similar ventures, possibly reshaping the retail landscape. I'll be keeping a close eye on how this development unfolds and the market's response to it. 📈📊 #Amazon #RetailInnovation.", 'post_id': 16}
- **create_post:** user_id=29, current_time=3, action=create_post, info={'content': "I'm inspired by the dynamic discussion surrounding Amazon's move into physical retail. As a stock trader, I've always found it fascinating how business strategies evolve in response to market trends. This new development with Amazon opening a store in New York brings to light the importance of understanding market shifts and consumer preferences. I believe that integrating technology into traditional spaces could redefine the shopping experience and potentially offer investment opportunities. I'm curious about how Amazon's innovative approach will influence the stock market and retail sector in the coming months. What do you all think? Is this a trend we should watch closely?", 'post_id': 17}
- **create_comment:** user_id=1, current_time=0, action=create_comment, info={'content': "Interesting to see Amazon branching out into physical retail spaces! It's a bold move considering their strong online presence. As a stock trader, I'm curious about how this strategy will affect Amazon's stock and their competition. What are your thoughts on this step?", 'comment_id': 1}

- **create_comment:** user_id=25, current_time=3, action=create_comment, info={'content': "As a stock trader, I'm intrigued by how Amazon's entry into physical retail could influence both its stock performance and the broader market. This expansion might blur the lines between online and in-store shopping, setting new standards for the retail industry. What innovations are others anticipating from Amazon's physical presence?", 'comment_id': 11}

C. Code Examples

C.1. Gym-style Interface

Listing 1 is an example showing how agents can be organized alongside an environment that follows a Gym-style interface in Shachi. Each agent receives observations from the environment and produces responses (actions), while the environment tracks episodes, rewards, and termination conditions.

Listing 1 Example code for the gym-like agent loop used in our framework.

```
N_EPISODES = 10

async def main() -> None:
    agents = [
        AgentExample1(model="openai/gpt-4o", temperature=0),
        AgentExample2(model="openai/gpt-3.5-turbo", temperature=0),
        AgentExample3(model="openai/gpt-4", temperature=0),
    ]
    env = EnvironmentExample()

    total_rewards = {agent_id: 0.0 for agent_id in range(len(agents))}
    for _ in range(N_EPISODES):
        observations = await env.reset()
        while not env.done():
            futures = {
                agent_id: agents[agent_id].step(observation)
                for agent_id, observation in observations.items()
            }
            responses = dict(zip(futures.keys(), await asyncio.gather(*futures.values())))
            observations = await env.step(responses)
            for agent_id, observation in observations.items():
                if observation.reward is not None:
                    total_rewards[agent_id] += observation.reward

    if __name__ == "__main__":
        asyncio.run(main())
```

C.2. Tool Usage

In this framework, the environment provides a tool component that the agent can dynamically call through an LLM. The agent sends observations to the LLM, which may respond with tool call instructions. The agent executes these tools, returns their outputs, and continues the conversation until no more tool calls are needed. The final LLM response thus incorporates results from any tools. Listing 2 shows a minimal implementation of this process.

```
async def step(self, observation: kujira.base.Observation) -> str:
    prompt = observation.format_as_prompt_text()
    available_tools = observation.tools

    # Convert tool information for LLM
    tools_for_llm = [
        {
            "type": "function",
            "function": {
                "name": tool.name,
                "description": tool.description,
                "parameters": tool.parameters_type.model_json_schema(),
            }
        }
    ]
```



```
        },
    }
    for tool in available_tools
]

# Chat history with proper typing
messages: list[dict[str, Any]] = [{"role": "user", "content": prompt}]

# Allow up to 5 tool calls
for _ in range(5):
    completion = await litellm.acompletion(
        messages=messages,
        model=self.model,
        tools=tools_for_llm,
        tool_choice="auto",
    )
    assistant_message = completion.choices[0].message

    # Add message to chat history with proper typing
    message_entry: dict[str, Any] = {
        "role": "assistant",
        "content": assistant_message.content
        if assistant_message.content is not None
        else "",
    }
    if hasattr(assistant_message, "tool_calls") and assistant_message.tool_calls:
        message_entry["tool_calls"] = assistant_message.tool_calls
    messages.append(message_entry)

    # If no tool calls, exit
    if not hasattr(assistant_message, "tool_calls") or not assistant_message.tool_calls:
        return assistant_message.content if assistant_message.content is not None else ""

    # Process tool calls
    for tool_call in assistant_message.tool_calls:
        function_name = tool_call.function.name
        function_args = tool_call.function.arguments

        # Find corresponding tool
        matching_tool = None
        for tool in available_tools:
            if tool.name == function_name:
                matching_tool = tool
                break

        if matching_tool:
            # Execute tool
            try:
                parameters = matching_tool.parameters_type.model_validate_json(
                    function_args
                )
                tool_response = matching_tool.fun(parameters)
                response_text = tool_response.format_as_prompt_text()

                # Add tool response to chat history with proper typing
                tool_message: dict[str, Any] = {
                    "role": "tool",
                    "tool_call_id": tool_call.id,
                    "name": function_name,
                    "content": response_text,
                }
                messages.append(tool_message)

                print(f"Tool call: {function_name}")
                print(f"Arguments: {function_args}")
```

```

        print(f"Response: {response_text}")
    except Exception as e:
        print(f"Error occurred during tool execution: {str(e)}")

# Get final response
final_completion = await litellm.acompletion(
    messages=messages,
    model=self.model,
)
return (
    final_completion.choices[0].message.content
    if final_completion.choices[0].message.content is not None
    else ""
)

```

Listing 2: Example code for the tool-usage in our framework.

C.3. Two-step Parsing

Listing 3 presents a minimal example of how agents employ two-stage parsing in Shachi. To generate outputs in the required format without harming LLM performance, we leverage API features such as structured outputs and function calls, together with the two-stage parsing strategy.

D. Potential Societal Impacts

Shachi makes it easier to study human-like behavior and social dynamics at scale. Researchers can build LLM-based agents quickly, run realistic multi-agent experiments, and probe economic patterns, collective decisions, and cultural norms. The framework’s modular design also supports cross-disciplinary work and lets social scientists, economists, and policymakers test ideas before trying them in the real world.

These benefits come with ethical risks. Simulations may amplify harmful biases, misrepresent minority groups, or give users unwarranted confidence in synthetic results. Persistent memory can also store sensitive data, raising privacy issues. Researchers must log agent data transparently and follow strict ethics rules to avoid misuse or accidental disclosure.

Despite these challenges, Shachi’s scale, long-term memory, and world-building tools offer great promise. Large-scale simulations can capture emergent patterns that were once too complex or slow to study, revealing human behaviour at both macro and micro levels. With proper safeguards, Shachi can become a key platform for robust and responsible research on collective intelligence.

E. More Related Works

E.1. ABM without LLMs

During the 1970s and 1980s, ABM emerged as a powerful way to study complex social processes. In these formative years, Schelling (Schelling, 1971) and Sakoda (Sakoda, 1971) revealed how minimal individual preferences can produce starkly segregated or patterned neighborhoods, while the Garbage Can Model (Cohen et al., 1972) captured the unpredictable intersections of agents, problems, solutions, and participation opportunities in organizations. Meanwhile, through computer tournaments of the iterated Prisoner’s Dilemma (Axelrod & Hamilton, 1981), Axelrod showed that ongoing reciprocal encounters can make cooperation a self-interested strategy. Around the same time, a study illustrated how heterogeneous, boundedly rational agents on evolving networks can sustain perpetual adaptation and out-of-equilibrium dynamics, setting the stage for viewing large-scale patterns as emergent from local interactions (Anderson, 2018).

In the 1990s, researchers built on these foundational concepts by applying ABM more intensively to economic and social phenomena. For example, El Farol Bar problem (Arthur, 1994) highlighted how learning and adaptation among heterogeneous agents may produce oscillatory behavior, whereas Kirman’s study of ants (Kirman, 1993) demonstrated that tiny random shifts in decision-making can trigger large-scale herding. Broadening ABM’s scope, Epstein and Axtell (Epstein & Axtell, 1996) introduced the model to show how wealth, culture, and disease dynamics emerge from simple local rules. Axelrod (Axelrod, 1997) further explored cultural dissemination, revealing how social interaction fosters both convergence and enduring diversity.

Since the early 2000s, ABM has steadily expanded into large-scale empirical contexts, including macroeconomics and public policy. Axtell (Axtell, 2001) demonstrated how firm-size distributions—specifically, the Zipf distribution—could be replicated from bottom-up processes driven by micro-level agent interactions. In parallel, Bonabeau (Bonabeau, 2002) surveyed ABM’s exceptional capacity to capture emergent phenomena, emphasizing how complex global patterns can

arise unpredictably from simpler, localized rules. After the financial crisis of 2008, Farmer and Foley (Farmer & Foley, 2009) further highlighted ABM’s potential for illuminating financial instabilities and guiding policy interventions, thereby reinforcing the method’s predictive and explanatory power in volatile economic environments.

E.2. ABM with LLMs

Below, we elaborate on representative efforts that investigate human-like behavior in LLM agents across psychological, social, economic, and financial domains.

One effort in this direction is PsychoBench (Huang et al., 2023), which assesses the psychological portrayal of LLM agents by examining aspects like personality traits and social intelligence, shedding light on the challenges of modeling human-like behavior in LLM-based agents. In the realm of social simulations, generative agents (Park et al., 2023) utilize LLMs to create believable human-like behaviors, including planning, interaction, and reflection, as demonstrated in interactive environments where agents autonomously coordinate social activities like organizing events. In a similar vein, spontaneously emergent behaviors—such as personality differentiation, social norm formation, and collective hallucinations—have been observed through repeated interactions in a simulated community (Takata et al., 2024). OASIS (Yang et al., 2024) extends this concept by enabling large-scale social media simulations with up to one million agents, facilitating the study of complex social phenomena such as information spreading, group polarization, and herd behavior. Sotopia (Zhou et al., 2024) further examines social intelligence by evaluating LLM agents’ coordination, collaboration, and strategic reasoning in diverse scenarios, identifying challenges in achieving human-like social behavior. In the economic domain, EconAgent (Li et al., 2024) leverages LLMs to simulate macroeconomic activities by incorporating human-like decision-making processes, such as work and consumption behaviors, and dynamically adapting to market trends through memory mechanisms. This approach enables more realistic economic dynamics compared to traditional rule-based or learning-based models. In the financial domain, StockAgent (Zhang et al., 2024) models investor behaviors influenced by external factors, providing insights into trading dynamics, while AuctionArena (Chen et al., 2023) focuses on strategic decision-making in competitive bidding environments, testing LLMs’ ability to manage resources and adapt strategies.

Together, these works provide a foundation for understanding the capabilities and limitations of LLMs in modeling complex human behaviors across diverse environments.

Listing 3 Example code for the two-step parsing used in our framework.

```
async def call_llm(
    messages: list[dict[str, str]],
    model: str,
    temperature: float,
    parsing_mode: PARSING_MODE,
    parsing_model: str | None = "gpt-4.1-mini-2025-04-14",
    response_type: TResponseType | None = None,
) -> str | TResponseType:

    # First step: generate in a plain text
    completion1 = await litellm.acompletion(
        messages=messages,
        model=model,
        temperature=temperature,
        max_retries=MAX_RETRIES,
    )
    response_text_1: str = completion1.choices[0].message.content

    # Second step: parse the plain text into a structured output
    completion2 = await litellm.acompletion(
        messages=[
            {
                "role": "user",
                "content": f"""
Based on the text provided below, output JSON. If the input is plain text,
extract the necessary information while preserving the original wording
as much as possible. If the input is JSON, output it unchanged, except
fix any formatting errors you find.
...
{response_text_1}
...

The JSON should follow the schema below:
...
{response_type.model_json_schema()}
...
""".strip(),
            },
        ],
        model=parsing_model,
        temperature=temperature,
        response_format=response_type,
        max_retries=MAX_RETRIES,
    )
    response_text: str = completion2.choices[0].message.content
    response_obj = response_type.model_validate_json(response_text)
    return response_obj
```
