Enabling User-Created Multi-Agent Simulations: Interactive and Customizable 2D Environments to Study Team Dynamics with LLM Agents

Mohammed Almutairi[1], **Charles Chiang**[1], **Haoze Guo**[1], **Nandini Banerjee**[1], **Svitlana Volkova**[2], **Daniel Nguyen**[2], **Tim Weninger**[1], **Michael Yankoski**[3], **Trenton W. Ford**[3], **Diego Gomez-Zara**[1†]

[1]University of Notre Dame, Notre Dame, IN, USA
[2]Aptima, Inc., Woburn, MA, USA
[3]William & Mary, Williamsburg, VA, USA

{malmutai, cchiang3, hguo3, nbanerje, tweninger, dgomezara}@nd.edu
{svolkova, dnguyen}@aptima.com
{myankoski, twford}@wm.edu

Abstract

Enabling users to create their own simulations offers a powerful way to study how environments shape agent behavior and intelligence. We introduce VIRT-LAB, a system that allows researchers and practitioners to design interactive, customizable simulations of team dynamics with LLM-based agents situated in 2D spatial environments. Unlike prior systems that restrict scenarios to predefined or static tasks, our approach empowers users to build environments, assign roles, and observe how agents coordinate, move, and adapt over time. A web-based interface makes these simulations accessible to both technical and non-technical users, supporting the design, execution, and analysis of complex multi-agent experiments without programming. By bridging team cognition behaviors with scalable agent-based modeling, our system provides a testbed for investigating how diverse environments influence coordination, collaboration, and emergent team behaviors. We demonstrate its utility by aligning simulated outcomes with empirical evaluations and a user study, underscoring the importance of customizable environments for advancing research on collective intelligence and adaptive agents.

1 Introduction

Understanding how teams perform, coordinate, and adapt in dynamic environments has long been central to advancing research on collaboration and decision-making [17, 15, 3]. With recent advances in large language models (LLMs), it is now possible to simulate team-like behaviors in increasingly complex tasks, where multiple agents can interact with one another and their surroundings in shared environments [27, 21, 26, 7]. However, current frameworks and systems remain limited: they often constrain researchers to fixed scenarios, lack support for spatial and temporal dynamics, and demand substantial technical expertise to configure or scale simulations effectively [11, 25].

To address these gaps, we present VIRT-LAB, a system that enables both technical and non-technical users to build their own simulations of team interactions with LLM-based agents. Through natural lan-

[†]Corresponding author

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Scaling Environments for Agents (SEA).

guage input and simple configuration settings, users can design scenarios in which agents collaborate, navigate 2D spaces, and adapt over time. The system supports customization of team composition, environment layouts, and task objectives, while providing a web interface for monitoring, managing, and analyzing simulation outcomes. By embedding agents in spatial and temporal contexts, VIRT-LAB extends agent-based modeling with richer dynamics and more accessible workflows.

The system's architecture combines a user-friendly front end with a simulation engine on the back end. Users can iteratively specify team members, tasks, and environmental entities via prompts, refine scenario elements, and observe the simulation unfold on a 2D map. The engine employs an event-scheduling mechanism to coordinate agents' actions in parallel, maintain shared states, and handle changes in the environment. This design abstracts away the technical complexity of coding and data management, allowing researchers to focus on scenario design and analysis.

Finally, we conducted evaluations to assess the system's performance and scalability, as well as a user study to examine how researchers and practitioners engage with the interface when designing their own simulations. These findings demonstrate the utility of VIRT-LAB both as an experimental platform and as a practical tool for studying team dynamics in customizable environments.

2 Related Work

Agent-Based Models (ABMs). Agent-based modeling has long been employed to simulate social systems and team processes [20]. These models are effective at capturing structured interactions, yet they typically rely on static, rule-based decision mechanisms to approximate human behavior [29, 4]. This reliance on fixed parameters often oversimplifies cognitive complexity, limiting the capacity to model adaptive, context-sensitive decision-making [11, 37, 19]. Recent work explores augmenting ABMs with machine learning to introduce more flexible behavior, but these approaches still struggle to capture rich communication or evolving social dynamics. LLMs provide a promising alternative: they can simulate nuanced conversations, memory, and belief updating, allowing agents to engage in more human-like interactions [8, 36]. Our system, VIRT-LAB, builds on ABM foundations while extending them into dynamic, interactive environments where LLM-driven agents can coordinate within spatial and temporal settings.

Multi-Agent LLM Frameworks. The rapid development of LLM-based agents has led to several multi-agent frameworks. Platforms such as OASIS [35], AutoGen [34], AgentSociety [28], and MindAgent [12] enable multi-party interaction and collaborative problem-solving. However, most of these systems emphasize dialogue and reasoning tasks without incorporating spatial navigation, temporal constraints, or dynamic environments that mirror real-world team challenges. Others, such as Generative Agents [26] or AgentCoord [25], include interfaces and multi-party simulations, but are tied to specific maps or domains, limiting flexibility. Furthermore, many frameworks require significant technical expertise to configure, which restricts their adoption among social scientists and practitioners. In contrast, VIRT-LAB provides a web interface and customizable 2D environments.

Evaluating Multi-Agent Systems and Team Behaviors. A growing body of research explores how to assess multi-agent systems, from benchmarks in cooperation and coordination to metrics for emergent social behavior [23, 7, 22]. These efforts highlight the difficulty of evaluating not only individual agent performance but also collective outcomes such as coordination, efficiency, and robustness. For team simulations, measures of coordination, adaptation, and role differentiation are particularly important, yet most existing frameworks provide limited support for such systematic evaluation. Our system integrates configurable evaluation metrics directly into the simulation pipeline, enabling users to analyze both process and outcome variables in team dynamics. In addition, we complement these automated metrics with a user study to better understand how researchers design, interact with, and interpret multi-agent simulations.

3 The VIRT-LAB Framework

3.1 Simulation setup

A team simulation in VIRT-LAB requires the user to create a *scenario* that outlines the team composition, which includes a set of *team members* with skills, attributes, and personality traits. The user

Table 1: Comparison between VIRT-LAB and existing state-of-the-art LLM-based multi-agent simulation frameworks

Framework	Web-UI	Customized simulation environment	Customized simulation scenario	Task-solving	Spatial Management
OASIS [35]	✗	✗	✓	✗	✗
Generative Agent [26]	✓	✗	✓	✗	✓
AutoGen [34]	✗	✗	✓	✓	✗
MindAgent [12]	✗	✗	✗	✓	✓
AgentCoord [25]	✓	✓	✗	✗	✗
AgentSociety [28]	✗	✗	✓	✗	✗
VIRT-LAB (Ours)	✓	✓	✓	✓	✓

must also describe the *environment*, including area names and entities residing in this environment. Lastly, the user must describe the *team goal* and the *metrics* to define their success (Figure 1).

Scenarios. VIRT-LAB asks the user to describe the scenario to be simulated. The description should include details about the team, its goals, the environment, and the maximum duration to run the simulation. Based on a conversation interface, the user provides details of the scenario, and VIRT-LAB might ask additional questions to infer details and create the scenario's elements. The system can also enhance the scenario by providing details and suggesting modifications to the user, who can accept or reject. As shown in Fig. 1a, a scenario could be *"Locate two individuals who have gone missing,"* which sets the "scene" for the simulation. Each scenario could also include information about the team composition to infer how many LLM-based agents need to be created for each team member. For example, the user can specify *"There are two searchers in this scenario."* The scenario's description might also include relevant entities that the team needs to employ or manage. In this example, VIRT-LAB needs to create missing individuals as entities to be controlled in the simulation. Lastly, VIRT-LAB will request information about the environment, including potential area names, spatial partitions, and layouts. The system displays the refined scenario prompt, where users can iteratively refine the elements and confirm the scenario to be run. The flexible way to build the scenario allows users to design a wide range of scenarios.

Agents. Based on the team described in the scenario, the user instantiates an LLM-based agent for each team member, populated with knowledge, characteristics, and attributes. Users can customize the agents' demographics, personality traits, psychological values, behavioral characteristics, and backstory memories. After analyzing the environment and the team objective, VIRT-LAB provides agents with initial contextual knowledge, including their roles, assigned objectives, the scenario's goal, and awareness of other agents participating in the simulation. VIRT-LAB will provide them contextual information of their physical environment and recent events through a sequence of conversations. Agent personality profiles and memory structures are encoded into embeddings and stored within a FAISS vector database [9]. To align agent behavior with predefined personalities, VIRT-LAB uses Retrieval-Augmented Generation (RAG) to retrieve relevant context—like short-term memories and traits via vector similarity searches on FAISS-stored embeddings.

Environment. VIRT-LAB represents the environment using 2D structures. The system generates the environment based on spatial metrics, such as width, length, and number of regions (i.e., rooms or areas). VIRT-LAB maps the 2D layout to a matrix $\mathcal{M}$, which encodes the traversability within regions. In this matrix representation, VIRT-LAB places traversable paths between walls, representing connections between spatial regions. This allows agents to determine specific paths for navigation between regions while enforcing the physical constraints of the environment. To determine the connectivity and semantic relationships between rooms, VIRT-LAB employs a graph representation from $\mathcal{M}$. The system generates a tree graph $\mathcal{G}$ in which leaf nodes $\mathcal{G} = \{g_1, g_2, \ldots, g_n\}$ correspond to a unique spatial region and their parent nodes represent a specific partition. Each node contains information about the specific region (e.g., name, characteristics), which is generated by the system. VIRT-LAB implements the graph $\mathcal{G}$ as an adjacency list of traversable spaces $\mathcal{C}$, representing connected rooms for the agents to transit.

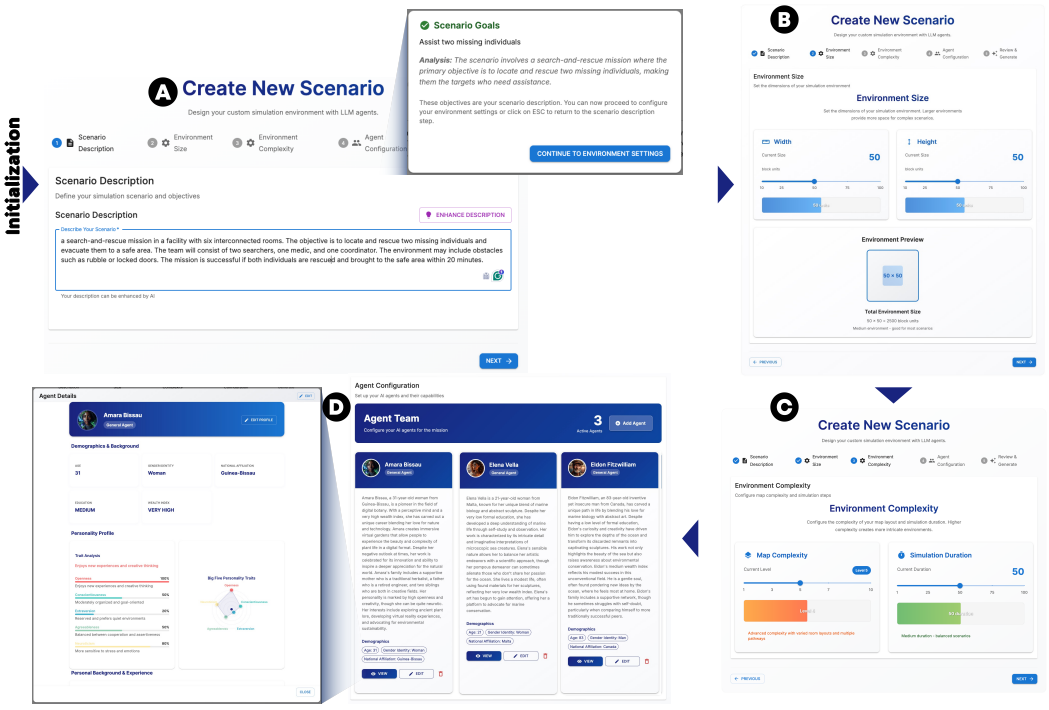

Figure 1: Initialization workflow of VIRT-LAB's web interface: **(A)** Scenario creation and objective validation; **(B)** Environment adjustment; **(C)** Team member configuration, including customized personalities, skills, and roles; **(D)** Setting the map complexity and simulation duration;

Entities. VIRT-LAB identifies and places entities in the environment based on the scenario's description. They might be located in a specific area if the scenario description includes the information, or be randomly located. Within each spatial region from $\mathcal{G}$, VIRT-LAB also locates entities based on the region's descriptions. VIRT-LAB categorizes entities into two main types: (a) *Interactive Entities*: Agents can interact with these entities to perform tasks during the simulation. VIRT-LAB asks the agents whether they want to make an action with the entity, and it will persist the changes on the environment; and (b) *Non-Interactive Entities*: These are entities that agents cannot manipulate, which provide contextual cues, and help agents navigate and understand the environment (e.g., walls).

3.2 Simulation execution

The VIRT-LAB simulation engine generates agent and environment events, and enables dynamic multi-party conversations and adaptive agent behavior (Figure 2).

Agent-Engine Interface. VIRT-LAB sends messages to each agent detailing their current location, the environment's current state, and results from their previous actions. With this information, agents can reason about their surroundings and decide on their next steps. VIRT-LAB then prompts each agent to make an action, communicate with another agent, or remain idle. If the agent chooses to act or communicate, VIRT-LAB schedules an *event* based on the agent's decision. Each event includes a duration in steps, the action (e.g., move, pick), contextual data, and the agents involved with the event.

Events. VIRT-LAB models the LLM agents' actions as discrete events that occur during the simulation. VIRT-LAB supports two types of events: *action* events and *communication* events. Action events affect the state of the environment, such as moving an agent to a new location or manipulating an entity. They can take a variable amount of time to complete, depending on the type of action and agent attributes. Communication events involve information exchanges between two or more agents. VIRT-LAB enables multi-turn interactions and requests the initiator agent to specify which agents should participate in the conversation. Once the conversation starts, VIRT-LAB

determines which agent should speak by predicting the most likely next speaker based on the ongoing conversation [34]. It also terminates the conversation once the information exchange becomes redundant. Agents have the choice to listen to the message or ignore it and continue with their current events.

Event Scheduling Manager. To manage the multiple events' resolutions and durations, VIRT-LAB employs an event scheduling manager that executes agents' events in the correct order. Events occur in parallel by advancing agents through incremental timesteps of their planned events, which enables agents to work on separate goals simultaneously. Before each execution, each event is validated by VIRT-LAB according to its knowledge of the main task and environment. This validation is done by asking an LLM to judge whether the agents' proposed action is reasonable and possible, given the current environment's state. Invalid events lead VIRT-LAB to re-prompt the agent, and valid events are sent to the event scheduling component.

At each time step t, VIRT-LAB retrieves the next scheduled event from a queue, executes the agent's actions, and updates the environment based on the agent's resolution. It records state changes, notifies other agents of the outcome, and allows them to update their knowledge and decide on future actions.

Navigation. To aid the agents' movement and exploration in the environment, VIRT-LAB provides directions to a specific room from their location. This design facilitates their spatial navigation. Agents can decide whether they follow the instructions or not.

3.3 Simulation Evaluation

The simulation ends when either VIRT-LAB detects that the main goal has been accomplished or the predefined simulation time expires. The user can obtain the scenario's final metrics and survey the agents to learn more from their experiences (Fig. 2).

Success Function. VIRT-LAB employs a success function that is generated by an LLM based on the scenario description. This enables simulation-specific evaluation criteria that accurately reflect each mission's objectives. The LLM generates a validation function that is executed at each simulation timestep to determine whether the mission has been successfully completed.

Post-hoc Survey. VIRT-LAB provides a modified Likert-based survey to the agents to capture their perspective on team performance and decision-making during the simulation. The survey also assesses the agents' ability to conceptualize the knowledge and viewpoints of other team members, offering insights into how the team is functioning. VIRT-LAB post-hoc interview process builds upon state-of-the-art methods to measure theory of mind (ToM) capabilities, which is the ability to infer, understand, and predict other agents' mental states relative to oneself. ToM is a broadly defined construct originating in developmental and cognitive psychology [32] that is now increasingly used to assess LLMs' abilities to perform logical and social reasoning [18]. VIRT-LAB integrates techniques to advance the measurement of various ToM facets that are relevant to the simulated teaming contexts.

Metrics. VIRT-LAB saves the final state of the environment in a structured JSON format, allowing for further analysis of agent performance, agent-environment interactions, and team dynamics. This log includes the time-stamped sequence of agent events, agents' intentions and rationale, and environment states. This log can be used for quantitative and qualitative analysis of agents' performance within the simulation.

3.4 Front-end Interface

VIRT-LAB provides an interactive web-based UI to facilitate user interaction, streamline scenario creation, and simplify result interpretation (Fig. 1 and 2). This interface enables users to easily define, configure, and refine simulation scenarios through an iterative process. Users can adjust parameters such as environment size, agent attributes (e.g., personality traits, roles, and skills), and map complexity, and preview their impact before execution. In the initial setup, the user provides the scenario description, the physical settings of the environment (i.e., width and height), and defines the map's complexity (i.e., number of regions). VIRT-LAB then interactively guides the user to provide further details and validate the proposed configuration. Once the scenario is defined, the user

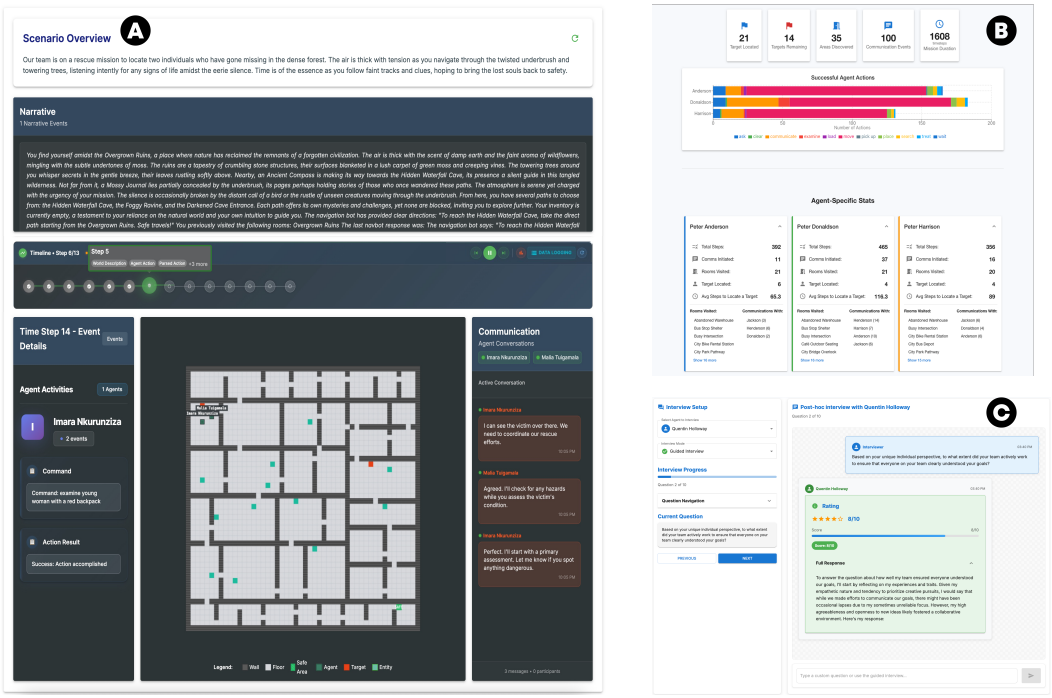

Figure 2: Execution workflow of VIRT-LAB's web interface: **(A)** Dashboard that streams the simulation narrative log, event timeline, and 2D map of agent movements; **(B)** An interview panel for probing agent reasoning; and **(C)** Summary of performance metrics and overall simulation result.

generates the agents, configuring their characteristics such as personality traits and roles. The system provides options to customize agents before starting the simulation, allowing experimentation with different team compositions. After the environment and the agents are configured, the system runs the simulation in real time on a 2D map, visualizing agent movements, decisions, and interactions as they unfold.

3.5 Implementation

We implemented the VIRT-LAB backend using Python and OpenAI GPT-4o-mini. We employed the `Comp-HuSim` framework to create the agents [10]. We generated the 2D environment using a binary partitioning algorithm. The environment and the entities were serialized and saved using the Pickle module[2]. This allowed us to store complex objects, such as the state of the environment, as binary data. We modeled the event scheduling system as a priority queue, which was implemented in Python as a heap. For the Web UI, we developed a user interface using *React* to enable visualization and interaction. The environment is rendered using Phaser[3], a game development library that supports 2D map representation and real-time updates.

4 Scalability Evaluation

To evaluate the scalability of VIRT-LAB, we conducted systematic stress tests under varying simulation conditions. Specifically, we examined the system performance when (i) increasing the environmental complexity and (ii) varying the number of agents per team. All experiments were carried out on a Linux server (Ubuntu 22.04) equipped with 125 GB RAM and an Intel(R) Core(TM) i7-7820X CPU. Below, we describe in detail the experimental variations used to assess scalability.

[2] `https://docs.python.org/3/library/pickle.html`
[3] `https://phaser.io/`

Environmental Complexity. We designed three environments of increasing difficulty by scaling the grid size to 30×30 (Low complexity), 40×40 (Medium complexity), and 50×50 (High complexity), and by randomizing the placement of 35 victims. As the environment size increased, agents were required to traverse longer distances and navigate more complex paths to locate and rescue victims, thereby increasing the computational demands on the system.

Team Size. We evaluated the system scalability with respect to team size by increasing the number of active agents per simulation, ranging from two to five agents per team. Each additional agent introduced more inter-agent interactions and decision-making processes, further contributing to system load and complexity. Each test configuration was executed three times to account for variability introduced by agent decision-making.

Measurements. Across all stress test variations, we evaluated the system using the following quantitative metrics. These measures were selected to capture the effectiveness, efficiency, and computational demands of the simulation system under different experimental conditions:

- **Number of Rescued Victims:** the total number of victims successfully rescued by the team during each simulation. A victim was considered rescued if it was located by an agent and transported to the designated hospital area within the simulation time limit.
- **Number of Action Events:** The total number of actions executed during the simulation, reflecting the dynamics of agent-agent and agent-environment interactions.
- **Number of Communication Events:** The total number of communication exchanges between agents during the simulation, including both dyadic and multi-agent conversations.
- **Simulation duration:** Total time step required to complete each simulation, measured from the start of agent deployment to the completion of the mission or the expiration of the simulation time step. To ensure consistency, the duration of the simulation was averaged across all runs for each configuration.

Results. Table 2 shows that VIRT-LAB scales with both team size and task complexity. On easy maps, all teams rescued all victims, but larger teams completed missions in fewer steps (from an average of 1,902.67 steps with two agents to 729.67 with five agents). As the map complexity increased, the mission duration grew and the success rates decreased for smaller teams (e.g., two agents on the hard map rescued on average 21 victims), whereas larger teams maintained a high number of rescued victims. These results demonstrate that VIRT-LAB can sustain effective simulation as both the complexity of the environment and the size of the team increase, highlighting its scalability under demanding conditions.

Table 2: Scalability results across map complexity levels and agent counts. Values are reported as mean $\pm$ standard deviation.

Environment Agents	Victims Rescued	Simulation Duration (Steps)	Action Events	Communication Events
Easy Map Complexity				
2 Agents	35	1,902.67 $\pm$ 84.98	1,431.0 $\pm$ 109.7	73.33 $\pm$ 9.03
3 Agents	35	1,309.67 $\pm$ 94.32	1,426.3 $\pm$ 91.2	69.67 $\pm$ 22.4
4 Agents	35	904.33 $\pm$ 40.66	1,262.3 $\pm$ 115.8	63.67 $\pm$ 9.1
5 Agents	35	729.67 $\pm$ 44.79	1,313.3 $\pm$ 80.9	61 $\pm$ 17.45
Medium Map Complexity				
2 Agents	33.33	$>$ 2,000	1,290.0 $\pm$ 12.3	29.67 $\pm$ 0.94
3 Agents	35	1,584.33 $\pm$ 51.91	1,842.3 $\pm$ 12.5	35.33 $\pm$ 18.19
4 Agents	35	1,184.67 $\pm$ 20.74	2,543.7 $\pm$ 53.5	32.33 $\pm$ 2.87
5 Agents	35	995.33 $\pm$ 20.89	2,665.3 $\pm$ 109.0	42 $\pm$ 12.03
Hard Map Complexity				
2 Agents	20.67	$>$ 2,000	1,380.7 $\pm$ 48.8	24.67 $\pm$ 10.14
3 Agents	31	$>$ 2,000	1,614.3 $\pm$ 35.7	26.33 $\pm$ 5.31
4 Agents	34	$>$ 2,000	1,583.3 $\pm$ 51.8	27.67 $\pm$ 4.78
5 Agents	35	1,686.33 $\pm$ 39.97	1,694.0 $\pm$ 7.3	31.33 $\pm$ 6.94

4.1 Usability Study

We conducted a laboratory usability study with 12 participants to evaluate the usability, interpretability, and perceived realism of the VIRT-LAB system. The usability study protocol was designed not to evaluate system performance, but to encourage user reflection on the ease of system use and the potential of LLM-powered team simulations in research settings. This sample size is supported by previous usability research [24]. University of Notre Dame's Institutional Review Board overview and approved this study (IRB Protocol Number #25-07-9459), and all participants signed an online consent form on Qualtrics prior to participation. Each session lasted 60 minutes, and participants were compensated with a $20 gift card for their time. The study addressed three primary research questions:

- **RQ1:** Can users use the VIRT-LAB system effectively and intuitively to create and configure team simulations as intended?
- **RQ2:** What benefits and limitations do users perceive when simulating human team dynamics using VIRT-LAB?
- **RQ3:** How do users' levels of simulation expertise influence their experiences when using VIRT-LAB?

Participants. We recruited participants from the simulation and team science communities through direct invitations, academic mailing lists, and social media posts. Our final sample consisted of participants from STEM-related fields, including researchers, graduate students, undergraduate students, and professionals with varying levels of simulation expertise. Based on self-reported experience with agent-based modeling and LLM agents, three participants were classified as experts, four participants as intermediate, and five participants as novices with little or no prior experience in these areas. To ensure accurate classification, two members of the research team reviewed the interview responses of the participants describing their previous experience with simulation systems.

Procedure. Each session took place either in person at a university research lab or virtually via Zoom. We conducted structured usability studies with nine participants, who first received a brief introduction and completed a background questionnaire on their experience with team simulations and AI systems. Sessions followed a semi-structured format: participants created short team simulation scenarios (e.g., *"A team of first responders battles the roaring wild. With the smoke thick in the air, they coordinate efforts to evacuate three residents"*) and then ran them using VIRT-LAB. If a scenario proved too complex for the allotted time, participants were encouraged to create a simpler alternative. During system use, participants were asked to think aloud, and afterwards they completed a post-study questionnaire and interview.

Measurements. We included several scales in the final survey to assess VIRT-LAB's realism and ease of use. The participants evaluated the *usability* of the system using the SUS scale [6] (Cronbach's α = 0.75). *Trust in the system* was assessed using a 5-point Likert scale adapted from [14] (α = 0.83). *Explanation satisfaction* was measured using items adapted from [14] that evaluated whether the explanations provided by the system were clear and supported the participants' understanding of the simulation results (α = 0.86). The *realism of the system and environment* was evaluated with items adapted from [33] that evaluated the participants' perceptions of the realism of the simulated environment (α = 0.71). *Perceived adoption* was measured with items adapted from [31] to evaluate participants' willingness to use the system in future research or professional contexts (α = 0.85). Additionally, we created a 5-point Likert scale with two items to evaluate the *realism of simulated agent and team behaviors*. The items included items such as *"The agents' behaviors were believable and consistent with their defined roles"* (α = 0.90). The questionnaire responses for each scale were averaged into a single score per participant's expertise level. Finally, open-ended questions were included to examine the perceived benefits and limitations of the participants using VIRT-LAB.

Results. Fig. 3 summarizes the results of the post-study questionnaire. In general, participants rated highly the usability of the VIRT-LAB system ($M = 3.8$), although differences were observed between the levels of experience. The novices reported a mean usability rating of $M = 3.8$ ($SD = 0.48$), while the intermediate users reported the highest usability rating of $M = 4.0$ ($SD = 0.33$). Expert users rated usability lower, with an average score of $M = 3.4$ ($SD = 0.75$). The level of trust in the

system varied. Among all participants, the average trust score was $M = 3.2$. The novices expressed a higher trust rating in the system ($M = 3.9, SD = 0.10$), while intermediates reported lower trust ratings ($M = 2.8, SD = 0.31$), and the experts gave the lowest trust scores ($M = 2.6, SD = 0.17$).

Satisfaction with the system's explanations was also mixed. The overall rating was $M = 3.7$, with novices reporting the highest satisfaction with an average $M = 4.0$ ($SD = 0.53$), intermediates giving average ratings ($M = 3.5, SD = 0.30$), and experts assigning the lowest scores ($M = 3.3, SD = 0.65$). The participants' perceptions of system and environment realism were similarly rated ($M = 3.6$) across all levels of experience. The novices reported an average realism rating of $M = 3.7$ ($SD = 0.16$). Expert users ($M = 3.4, SD = 0.14$) and intermediate users ($M = 3.4, SD = 0.12$) reported lower scores. The perceptions of system and environment realism were assessed as moderate across the various groups, with an overall mean rating of $M = 3.6$. Novices rated realism higher ($M = 3.7, SD = 0.16$), while intermediates ($M = 3.4, SD = 0.12$) and experts ($M = 3.4, SD = 0.14$) gave similar, lower ratings.

Participants' willingness to adopt the VIRT-LAB system in future use followed a similar pattern. On average, perceived adoption was rated at $M = 3.5$. Novices expressed the highest adoption intentions ($M = 4.0, SD = 0.37$), while intermediates provided average ratings ($M = 3.2, SD = 0.17$), and experts reported the lowest adoption scores ($M = 3.0, SD = 0.45$). Finally, perceptions of agent and team behavior realism were also mixed. The overall average rating was $M = 3.3$. Novices rated realism high ($M = 3.6$), intermediates at $M = 3.25$, and experts at $M = 3.0$, with little variation within each group.

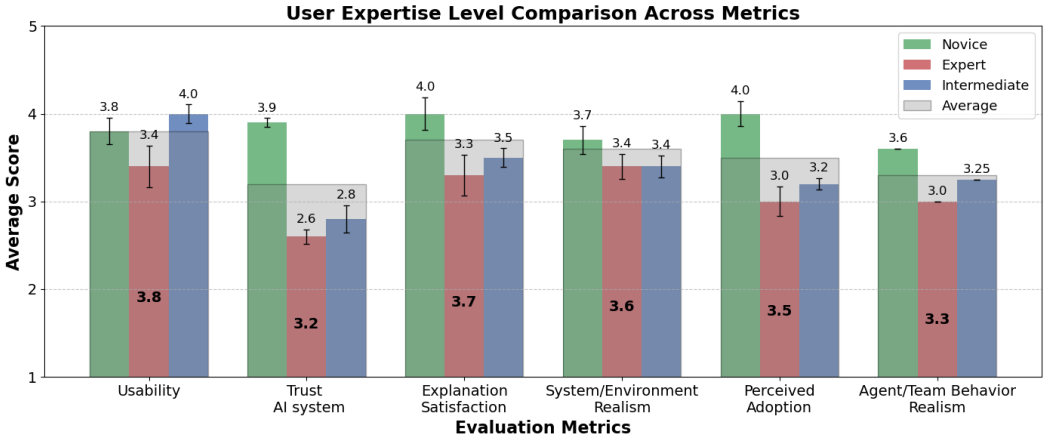

Figure 3: Participants' survey responses. Error bars denote standard errors. Average score per metric is displayed in the middle.

Interviews The researchers transcribed recordings of participants' task interactions and interviews, then collaboratively developed an initial codebook. They independently coded the remaining data, resolving differences through discussion until reaching consensus. Codes were grouped into themes, focusing on usability, realism, and simulation control—using Braun and Clarke's thematic analysis approach [5]. We report the following key findings:

KF1: VIRT-LAB*'s workflow felt intuitive, though expectations varied with prior experience.* Most participants found the simulation workflow process aligned with how they naturally think about creating a simulation. As P(01) explained, *"I do like this workflow . . . I just need to follow each step"*. Many participants reported that VIRT-LAB lowered the barrier to trying many scenario variants and observing different team behaviors in action. (P06) "*No coding is great— it [the system] lets me set up variants fast and focus on the team behavior, type it and wait 10 minutes, whatever it took to, like, fully do it. I mean, I couldn't have coded that in 10 minutes ... that is a huge pro. Even when I knew NetLogo pretty well, I don't think I could have done that in 10 minutes*". The visualization helped users iterate and compare alternatives. (P11) "*Seeing it play out on the map—almost like a video game—made it easier to explore different [simulation].*" Nevertheless, participants pointed the limitations that impacted realism and control. (P03) "*The [agent] interview answers felt a bit*

'GPT-like' and didn't always match what they did". Similarly, (P01) explained, *"I was expecting like something real human [would] say ..., but [it] will help a lot, I like the idea a lot to talk to them."*

KF2: *Lack of* VIRT-LAB*'s transparency reduced users' trust in the simulations.* Across expertise levels, participants questioned what data was sent to the LLM, what was returned, and how these exchanges shaped the simulation outcomes, concerns that directly influenced their trust in the system. Novices and intermediates explicitly linked clarity to trust. (P03) noted, *"if this part can be transparent, I think my trust... [goes] up"*, while also reporting confusion when they could not tell what the system was doing. These users looked for signs *"what's happening now"* (P03) signals, expressing unease when backend processes felt delayed. As (P07) remarked, *"I don't know what the [system] is doing. It's kind of like an in-house system, but I think it's pretty common"*. Experts approached trust more analytically, focusing on how data were collected and processed by the LLM and whether simulated agent behavior aligned with assigned traits. As (P04) reported, *"I would trust the agents more if they tied their responses to actual events rather than generic personality templates."*

5 Limitations and Future Work

While VIRT-LAB offers a flexible platform for simulating and studying human team dynamics through LLM-based agents, our study has limitations that need to be addressed in the future. First, VIRT-LAB inherits constraints of the underlying LLMs. As with all LLM-based systems, our agents may reproduce societal biases embedded in model training data [2], which may affect the behavioral fidelity. Our current results use general-purpose LLMs without extensive fine-tuning for domain-specific applications or agent-specific roles. Future work may explore the integration of domain-adapted LLMs and assess how model selection impacts emergent team behaviors. Second, scaling to larger teams and groups (e.g., 10, 50, 100) remains expensive, as frequent LLM calls increase token usage and processing time. Improvements, such as using local LLMs [13, 30, 1] or optimization techniques [16], are important. Third, testing this system with other LLM models is important to assess the differences produced by the specific models. Future evaluations should employ different models and assess where differences can occur, especially in conversations among agents. Fourth, direct feedback on VIRT-LAB's interactive interface from a diverse range of end users, such as social scientists and agent-based modeling experts, remains limited. Future usability studies should involve a large and more diverse group of participants representing various domains. Finally, future work should include ablation study to examine the contribution of each system component to agent performances and behavioral realism.

6 Conclusion

This paper introduced VIRT-LAB, a user-friendly, customizable, and scalable simulation system for modeling teams in complex environments. It supports multi-agent interactions, spatial reasoning, surveys, and automated evaluation. With its front-end interface, researchers can design and analyze team simulations without coding expertise. By lowering technical barriers, VIRT-LAB provides new opportunities for studying team dynamics through tailored scenarios, multi-party dialogue, and spatial simulations.

Acknowledgments and Disclosure of Funding

This work was partially supported by the Defense Advanced Research Projects Agency (DARPA) under Agreements HR00112490408, HR00112490410, and HR00112430361; Alfred Sloan Foundation Award G-2024-22427; National Science Foundation under Grant Number 2317987; and the Microsoft Accelerating Foundation Models Research (AFMR) grant program. We also thank Matthew Belcher for his contributions to the software development that supported the implementation of our experimental system.

References

[1] Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv*

preprint arXiv:2412.08905, 2024.

[2] Suhaib Abdurahman, Mohammad Atari, Farzan Karimi-Malekabadi, Mona J Xue, Jackson Trager, Peter S Park, Preni Golazizian, Ali Omrani, and Morteza Dehghani. Perils and opportunities in using large language models in psychological research. *PNAS nexus*, 3(7):pgae245, 2024.

[3] Mohammed Almutairi, Charles Chiang, Yuxin Bai, and Diego Gomez-Zara. taifa: Enhancing team effectiveness and cohesion with ai-generated automated feedback. In *Proceedings of the 4th Annual Symposium on Human-Computer Interaction for Work*, pages 1–25, 2025.

[4] Li An, Volker Grimm, Abigail Sullivan, BL Turner Ii, Nicolas Malleson, Alison Heppenstall, Christian Vincenot, Derek Robinson, Xinyue Ye, Jianguo Liu, et al. Challenges, tasks, and opportunities in modeling agent-based complex systems. *Ecological Modelling*, 457:109685, 2021.

[5] Virginia Braun and Victoria Clarke. Using thematic analysis in psychology. *Qualitative research in psychology*, 3(2):77–101, 2006.

[6] John Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[7] Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chen Qian, Chi-Min Chan, Yujia Qin, Yaxi Lu, Ruobing Xie, et al. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents. *arXiv preprint arXiv:2308.10848*, 2(4):6, 2023.

[8] Yun-Shiuan Chuang, Agam Goyal, Nikunj Harlalka, Siddharth Suresh, Robert Hawkins, Sijia Yang, Dhavan Shah, Junjie Hu, and Timothy T Rogers. Simulating opinion dynamics with networks of llm-based agents. *arXiv preprint arXiv:2311.09618*, 2023.

[9] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. *arXiv preprint arXiv:2401.08281*, 2024.

[10] Chengyu Fan, Zaynab Tariq, Nafis Saadiq Bhuiyan, Michael G Yankoski, and Trenton W Ford. Comphusim: Persistent digital personality simulation platform. In *Adjunct Proceedings of the 32nd ACM Conference on User Modeling, Adaptation and Personalization*, pages 98–101, 2024.

[11] Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1–24, 2024.

[12] Ran Gong, Qiuyuan Huang, Xiaojian Ma, Yusuke Noda, Zane Durante, Zilong Zheng, Demetri Terzopoulos, Li Fei-Fei, Jianfeng Gao, and Hoi Vo. MindAgent: Emergent gaming interaction. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3154–3183, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.200. URL `https://aclanthology.org/2024.findings-naacl.200/`.

[13] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[14] Robert R Hoffman, Shane T Mueller, Gary Klein, and Jordan Litman. Metrics for explainable ai: Challenges and prospects. *arXiv preprint arXiv:1812.04608*, 2018.

[15] Seung Wan Hong, Davide Schaumann, and Yehuda E Kalay. Human behavior simulation in architectural design projects: An observational study in an academic course. *Computers, Environment and Urban Systems*, 60:1–11, 2016.

[16] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[17] Stephen E Humphrey and Federico Aime. Team microdynamics: Toward an organizing approach to teamwork. *Academy of Management Annals*, 8(1):443–503, 2014.

[18] Michal Kosinski. Evaluating large language models in theory of mind tasks. *Proceedings of the National Academy of Sciences*, 121(45):e2405460121, 2024. doi: 10.1073/pnas.2405460121.

[19] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40:e253, 2017.

[20] Samuel Lapp, Kathryn Jablokow, and Christopher McComb. Kaboom: an agent-based model for simulating cognitive style in team problem solving. *Design Science*, 5:e13, 2019.

[21] Yuan Li, Yixuan Zhang, and Lichao Sun. Metaagents: Simulating interactions of human behaviors for llm-based task-oriented coordination via collaborative generative agents. *arXiv preprint arXiv:2310.06500*, 2023.

[22] Jiaju Lin, Haoran Zhao, Aochi Zhang, Yiting Wu, Huqiuyue Ping, and Qin Chen. Agentsims: An open-source sandbox for large language model evaluation. *arXiv preprint arXiv:2308.04026*, 2023.

[23] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*, 2023.

[24] Jakob Nielsen and Thomas K Landauer. A mathematical model of the finding of usability problems. In *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*, pages 206–213, 1993.

[25] Bo Pan, Jiaying Lu, Ke Wang, Li Zheng, Zhen Wen, Yingchaojie Feng, Minfeng Zhu, and Wei Chen. Agentcoord: Visually exploring coordination strategy for llm-based multi-agent collaboration. *arXiv preprint arXiv:2404.11943*, 2024.

[26] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22, 2023.

[27] Joon Sung Park, Carolyn Q Zou, Aaron Shaw, Benjamin Mako Hill, Carrie Cai, Meredith Ringel Morris, Robb Willer, Percy Liang, and Michael S Bernstein. Generative agent simulations of 1,000 people. *arXiv preprint arXiv:2411.10109*, 2024.

[28] Jinghua Piao, Yuwei Yan, Jun Zhang, Nian Li, Junbo Yan, Xiaochong Lan, Zhihong Lu, Zhiheng Zheng, Jing Yi Wang, Di Zhou, et al. Agentsociety: Large-scale simulation of llm-driven generative agents advances understanding of human behaviors and society. *arXiv preprint arXiv:2502.08691*, 2025.

[29] William Rand and Christian Stummer. Agent-based modeling of new product market diffusion: an overview of strengths and criticisms. *Annals of Operations Research*, 305(1):425–447, 2021.

[30] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

[31] Viswanath Venkatesh and Hillol Bala. Technology acceptance model 3 and a research agenda on interventions. *Decision sciences*, 39(2):273–315, 2008.

[32] Henry M. Wellman. Theory of mind: The state of the art. *European Journal of Developmental Psychology*, 15(6):728–755, 2018. doi: 10.1080/17405629.2018.1435413.

[33] Bob G Witmer and Michael J Singer. Measuring presence in virtual environments: A presence questionnaire. *Presence*, 7(3):225–240, 1998.

[34] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 2023.

[35] Ziyi Yang, Zaibin Zhang, Zirui Zheng, Yuxian Jiang, Ziyue Gan, Zhiyu Wang, Zijian Ling, Jinsong Chen, Martz Ma, Bowen Dong, et al. Oasis: Open agents social interaction simulations on one million agents. *arXiv preprint arXiv:2411.11581*, 2024.

[36] Jintian Zhang, Xin Xu, Ningyu Zhang, Ruibo Liu, Bryan Hooi, and Shumin Deng. Exploring collaboration mechanisms for llm agents: A social psychology view. *arXiv preprint arXiv:2310.02124*, 2023.

[37] Wei Zhang, Andrea Valencia, and Ni-Bin Chang. Synergistic integration between machine learning and agent-based modeling: A multidisciplinary review. *IEEE Transactions on Neural Networks and Learning Systems*, 34(5):2170–2190, 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]

2. **Limitations**

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

   Answer: [Yes]

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: This paper does not include theoretical results.

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: This paper introduces an architecture and interactive system. We provide details to explain clearly and fully how it operates.

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 a repository with the simulation files and data analysis scripts for reproducibility purposes. `https://anonymous.4open.science/r/SEA-2025-Submission-D5E1/`

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]

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]

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]

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]

10. **Broader impacts**

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

    Answer: [NA]

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]

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]

13. **New assets**

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

    Answer: [NA]

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: [Yes]

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: [Yes]

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: [NA]