Large Language Models Need Consultants for Reasoning: Becoming an Expert in a Complex Human System Through Behavior Simulation

Anonymous ACL submission

Abstract

Large language models (LLMs), in conjunction with various reasoning reinforcement methodologies, have demonstrated remarkable capabilities comparable to humans in fields such as mathematics, law, coding, common sense, and world knowledge. In this paper, we delve into the reasoning abilities of LLMs within complex human systems. We propose a novel reasoning framework, termed "Mosaic Expert Observation Wall" (MEOW) exploiting generative-agents-based simulation technique. In the MEOW framework, simulated data are utilized to train an expert model concentrating "experience" about a specific task in each independent time of simulation. It is the accumulated "experience" through the simulation that makes for an expert on a task in a complex human system. We conduct the experiments within a communication game that mirrors real-world security scenarios. The results indicate that our proposed methodology can cooperate with existing methodologies to enhance the reasoning abilities of LLMs in complex human systems.

1 Introduction

Large language models (LLMs) are evolving rapidly. Through extensive training on vast datasets, they exhibit remarkable open-domain reasoning capabilities. Llama-2 (Touvron et al., 2023) shows superior reasoning performance across various evaluation benchmarks, including mathematics (Cobbe et al., 2021; Hendrycks et al.), coding (Chen et al., 2021), common sense (Bisk et al., 2020; Sap et al., 2019), world knowledge (Kwiatkowski et al., 2019), and reading comprehension (Rajpurkar et al., 2018; Choi et al., 2018). Techniques such as fine-tuning (FT) (Hu et al., 2021; Roziere et al., 2023), chain-of-thought (CoT) (Kojima et al., 2022), in-context learning (ICL) (Wei et al., 2022), and retrieval-augmented generation (RAG) (Lewis et al., 2020) further enhance the performance of LLMs in specific tasks or domains.

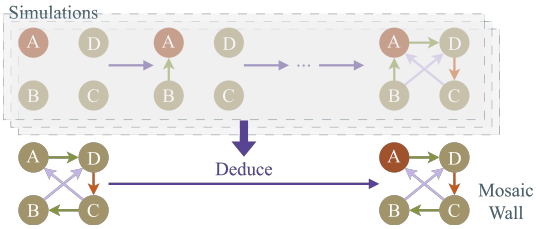

Figure 1: Mosaic investigation wall is what detectives use to visualize clues and interactions within cases. Through simulating current individuals' behaviors and interactions on a Mosaic wall, a detective can deduce who may be the crime in corresponding real case.

However, these methodologies may not deal well with complex human strategic decision-making and interaction scenarios. For instance, in the context of financial security, due to privacy issues, there is a lack of data and explicit knowledge to implement FT and RAG effectively. Moreover, ICL combined with CoT often leads LLMs to stick to examples or historical information when solving problems. In fact, this approach is not always effective, as financial attackers may not repeat previously detected or blocked methods to attack the financial system, and even some valid behaviors by the normal can lead to systemic risk (Eisenberg and Noe, 2001).

Simulation techniques serve as indispensable tools for tackling challenges within complex human systems. An inherent feature of simulation is its capacity to delve into interactions among individuals, offering an optimal solution to problems related to complex systems. Studies across diverse fields (Cui et al., 2010; Zhou et al., 2020; Hui et al., 2022) have validated their effectiveness. With the advent of generative agents technique (Park et al., 2023) which realized individual-level human behavior simulation in a sandbox, we believe that simulation techniques based on it can indeed extend beyond complex natural systems to complex human systems.

In this paper, after utilizing the generative-

agents-based simulation technique to simulate a complex human system, we design a novel framework, "Mosaic Expert Observation Wall" (MEOW), which imitates a detective simulating and analyzing a case through Mosaic investigation wall as illustrated in Figure 1. In MEOW, real game data are processed by an expert model trained on simulated data and converted into natural language prompts as expert observation to assist LLM reasoning. From the perspective of assisted LLM agent, the expert model serves as its consultant. This novel methodology addresses aforementioned problems of existing LLM reasoning reinforcement methodologies, which offers the potential to improve LLM reasoning independently or in conjunction with them in complex human systems. As a validation, we conduct the experiments in a Werewolf-like communication game where players infer others' identities based on common sense and game strategies. This scenario is a simplified representation of real-world security scenarios involving conscious attackers and involuntary negative behaviors. Subsequently, another LLM-based agent acts as a detective to infer each player's identity.

MEOW addresses challenges in analyzing problems in complex human systems and is capable of collaborating with existing LLM reasoning reinforcement approaches. Our experimental results demonstrate the effectiveness of MEOW. Additionally, through these experiments, we stumble across and summarize some serious challenges of applying MEOW to more complex human systems.

2 Related Work

2.1 LLM Reasoning

When applying a general LLM to a specific scenario, there is often a demand for focusing on knowledge relevant to that scenario. To meet this demand, methodologies such as FT, CoT, ICL, and RAG have been proposed and are widely used. Among them, FT is the most powerful one. Efficient fine-tuning methods based on Low-Rank Adaptation of LLMs (LoRA) (Hu et al., 2021) have enabled the realization of specialized LLMs for code (Roziere et al., 2023), biomedical applications (Tinn et al., 2023), and more. CoT (Kojima et al., 2022), originating from a special prompt, "Let's think step by step", is a data-free approach to improve LLM reasoning on complex problems. CoT methods, and further tree-of-thought (Yao et al., 2023) feature using special prompts and context organization modes to direct the LLM in generating a series of steps to solve a problem. Since LLMs require context to filter candidate tokens, more sufficient and meaningful contexts enhance their ability to reason about complex problems. ICL, which makes LLMs learn from analogy, was first applied to solve mathematical problems (Wei et al., 2022). Through prompts of problem-solving examples, the LLM imitates the correct steps, making it a kind of manual CoT based on example data (Dong et al., 2022). RAG, leveraging external professional knowledge, can effectively mitigate the hallucination of LLMs (Lewis et al., 2020). By retrieving external knowledge, LLM applications like Copilot [1] and GPT-4 (Achiam et al., 2023) can generate and reason more accurately. However, FT and RAG require extremely huge amounts of high-quality data that are often unavailable in complex human systems. While CoT and ICL require less data, the complexity of human systems makes it challenging to solve problems using analogical and simple "step by step" patterns. Therefore, none of these methodologies is good enough to analyze complex human systems at present.

2.2 Generative-agents-based Simulation

Generative-agents-based simulation is a form of agent-based simulation whose agents are text-based LLMs instead of typical deep learning (DL) models trained in closed domains. Proposed by Park et al. (2023), this simulation leverages the strengths of generative agents, encompassing common sense reasoning, contextual retrieval, and humanity. With these advantages, logical thoughts and human-like social behaviors have been observed during the simulation in a sandbox. Furthermore, Gao et al. (2023) developed a social network simulation framework building upon the work of generative agents. Their experiments demonstrated that both individual-level and population-level simulations within the generative agents social network had remarkable alignment with data of information, emotion, and attitude propagation in real world.

3 Behavior Simulation in Communication Game

3.1 Game Rules

The communication game under discussion in this study is called "Find The Spy". our experiments are based on a four-player version of this game. In

[1] Copilot: https://copilot.microsoft.com/

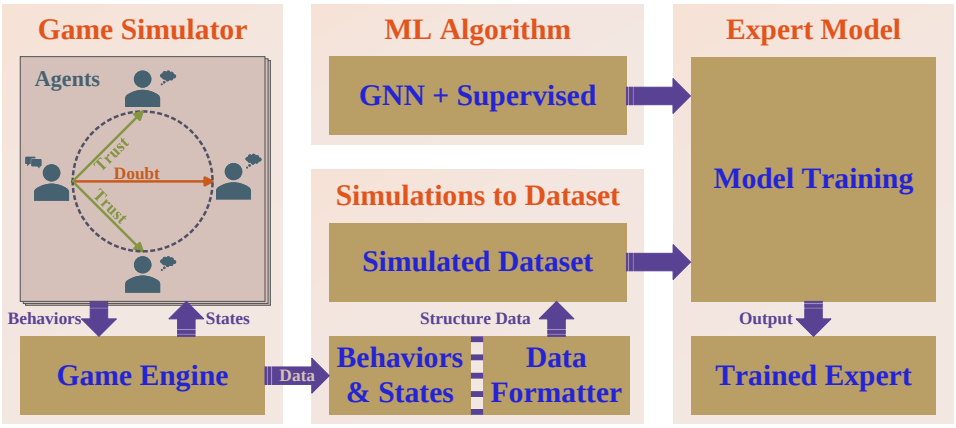

Figure 2: The illustration of implementing generative-agents-based simulation and training an expert model based on simulated data in the four-player version of "Find The Spy".

this scenario, players are divided into two groups: "folk" and "spies". Three players belong to the "folk" group, while one player is designated as a "spy". At the beginning of the game, "folk" players receive the same word, while the "spy" receives a different word. These two words, while distinct, share some commonalities (e.g. two words, "apple" and "pineapple", are both fruits). Each player only knows his own word and remains ignorant of both his identity ("ordinary people" belongs to "folk" or "spy" belongs to "spies") and the identities of the other players.

The objective of the game is to eliminate the opposing group of players through communication and voting. For "spies", the goal is to conceal their identity to avoid being voted out. In the four-player scenario, when only two players remain and one of them is the "spy", the referee announces the "spies" as the winners. Conversely, the "folk" players aim to identify the "spies" based on the communication and vote them out through the voting process. If all remaining players are "folk", the referee announces the "folk" as the winners.

The detailed game processes are documented in Appendix A.

3.2 The Scope of Behavior Simulation

Due to the features of communication games, players are required to possess basic common sense, reasoning abilities, and expressive skills. Tiny differences in the comprehension and analyses of utterances may lead a game to another result. It is these uncertain factors that render the game engaging and unpredictable, thereby constituting what we refer to as a complex human system. As the essence of complex systems, neither closed-form expressions nor accurate distributions about future states are feasible due to the chaotic interactive relationships among individuals in such systems (Lorenz, 1963; May, 1976). Consequently, there is no fixed distribution of individual behaviors and states in a complex human system, implying that an increased volume of data will facilitate a more comprehensive analysis of this system, if the data differ from existing historical data. To acquire such data in a complex human system, direct observation of the real world is the optimal method, but often costly, inefficient, time-consuming, and even unavailable. Under such circumstances, resorting to simulated data emerges as an alternative. DL generative models have proven their effectiveness in supplementing image datasets for better training (Shrivastava et al., 2017). However, relying exclusively on the outputs of DL models may not be effective in a complex human system. On the one hand, images within a certain dataset follow a relatively stable distribution compared to behaviors and states in a complex human system, meaning that supplementing the data with data of the same distribution contributes to a specific image dataset but not to a complex human system. On the other hand, the complexity of behaviors and states, coupled with limited real-world historical data, makes it challenging to train a DL model. As a result, behavior simulation currently stands as the only method to extend or even create data to assist analyzing a complex human system. Fortunately, with LLMs, it is feasible to create a generative agent capable of playing this game with sufficient knowledge as behavior simulation in the game system.

3.3 Instance in "Find The Spy"

Specifically for the "Find The Spy" game, we design an LLM-driven judge agent tasked with identifying the "spy" through players' utterances and votes, relying solely on knowledge acquired from pre-training. Simultaneously, given that no historical data are available, we utilize behavior simulation to create simulated "Find The Spy" game datasets, which are used to train an expert model that predicts the "spy" player. We name this framework the "Mosaic Expert Observation Wall" (MEOW). In MEOW, the simulated data, rich in agents' interaction behaviors and states, constructs multiple graph data as mosaic investigation walls. Expert machine learning (ML) models, trained on these data, provide the judge agent with expert observations on the real system to refine the analyses. We anticipate that the judge agent's performance in identifying the "spy" will improve after consulting the expert models. If such improvements are observed, it indicates that LLM agents can reason more effectively in complex human systems through simulation, a data-efficient method besides CoT and FT.

4 Architecture and Methodology

To implement MEOW in the communication game, as illustrated in Figure 2, a game simulator is required to perform a specified number of simulations. Subsequently, the simulated data of behaviors and states are converted from text into graph-structured data. Finally, expert models are trained on the data and labels, using the chosen model and algorithm. In the following subsections, we will elaborate on the construction of the game simulator and the detailed implementation of MEOW.

4.1 Game Simulator

Our game simulator primarily comprises two components: generative agents and a game engine. The generative agents, driven by multi-turn dialogues, generate their analyses, utterances, and behaviors, including tendencies and votes, in text form. These behaviors are converted into dictionary-format data, which are then inputted into the game engine to represent interactions among agents. Subsequently, the game engine processes these interactions and updates the states of the system and agents in accordance with the game rules. Finally, upon the conclusion of the game, all behaviors and states data are collected and reformatted as the output of the simulator.

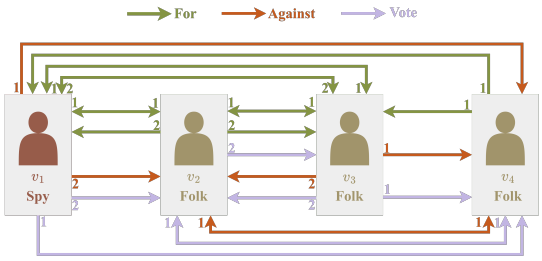

Figure 3: Example of heterogeneous graph. The numerical labels i at the tail of each arrow represent the ith round of the game.

4.1.1 LLM Agent

A generative agent takes action based on its memory, planning, retrieval, and reflection (Park et al., 2023). In the "Find The Spy" game, our player agents are designed in a similar paradigm. We utilize CoT prompt templates to guide the inference of player agents during different phases of the game. We divide the game into three processes and six phases, which are described in detail in Appendix B. The complete prompts along with examples of transactions will soon be available online.

4.1.2 Game Engine

The game engine primarily serves two functions. The first function involves broadcasting open information. Given that LLM agents heavily rely on context, particularly in multi-turn dialogue role-play scenarios, we incorporate confirmation responses to ensure that the agents receive the broadcast successfully. For instance,

```
User: (system)(Player Bob is eliminated.)
Assistant: Okay, I see.
```

This context is manually added instead of being generated by the LLM.

The second function of the game engine is to update the states of the system and agents, such as counting votes and eliminating. Concurrently, it records the utterances and behaviors in files as the simulator output. These files are subsequently transformed during the implementation of MEOW.

4.2 Implementation of MEOW

Once acquiring the simulated data, the initial steps involve transforming this data into graph-structured representations and generating datasets. Utilizing these datasets, we train two expert models. During the inference phase of the actual game, the judge agent identifies the "spy", initially depending on

raw record files, and then the inference is refined with expert observations from the trained expert models.

4.2.1 Simulated Data to Heterogeneous Graph

The simulated data are transformed into heterogeneous graphs, as depicted in Figure 3. Each game corresponds to a directed heterogeneous graph, denoted as $G = (V, E)$, where V represents the set of nodes and E denotes the set of directed edges. The set V comprises a single type of node, i.e.,

$$V = \{v_i|(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{n}, \mathbf{x}_i \in \mathbb{R}^{1\times 4}, \mathbf{y}_i \in \{0, 1\},$$

where each node v_i corresponds to a player and n signifies the total number of players (in our experiments, $n = 4$). Node v_i is characterized by a 4-dimensional one-hot tensor feature $\mathbf{x}_i$ and a label $\mathbf{y}_i$ indicating its role (0 for "folk" and 1 for "spy"). One game consists of either one or two rounds. The set of directed edges are denoted as:

$$\begin{aligned} E_1 &= E^1_{for} \cup E^1_{against} \cup E^1_{vote}, \\ E_2 &= E_1 \cup E^2_{for} \cup E^2_{against} \cup E^2_{vote}, \\ E_1, E_2 &\subseteq V \times V, \end{aligned}$$

where E^i_{for}, $E^i_{against}, E^i_{vote}$ represent the trusting, doubting, and voting interactions of the top i rounds respectively, if the round exists.

Denoting $\mathbf{G}$ as the set of all the games and $\mathbf{G}_\lambda$ as the set of games where "spy" is not eliminated in the first round. Then, we create two datasets D_1, D_2 respectively containing the first round of each game in $\mathbf{G}$, and two rounds of partial games in $\mathbf{G}_\lambda$.

$$\begin{aligned} D_1 &= \{G_k|G_k = (V^k, E^k_1), G_k \in \mathbf{G}\}, \\ D_2 &= \{G_l|G_l = (V^l, E^l_2), G_l \in \mathbf{G}_\lambda\}. \end{aligned}$$

4.2.2 Expert Model Design

We formulate the problem as a four-class classification task, where one out of the four players is the "spy". To address this problem, we introduce two expert models respectively trained on D_1 and D_2. These models share a similar architectural design, as illustrated in Figure 4. We take the first expert model as an example to illustrate its design.

The model employs two layers of the Graph Attention Network v2 (GATv2) (Brody et al., 2021). In the one-round graph that we construct, there are three distinct types of edges. GATv2 enables isolated message passing on different types

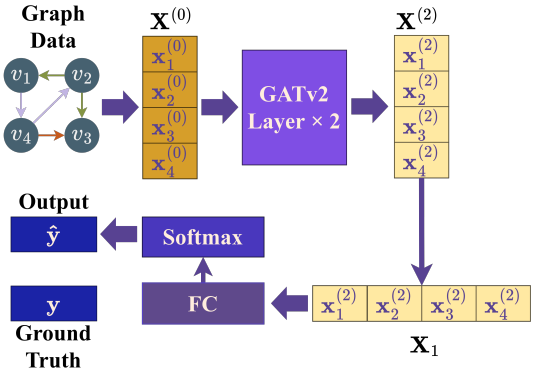

Figure 4: The architecture of expert models.

of edges. The input node features matrix is denoted as $\mathbf{X} = [\mathbf{x}_1 \quad \mathbf{x}_2 \quad \mathbf{x}_3 \quad \mathbf{x}_4]^{\mathbf{T}}$, the graph connectivity information as $\mathbf{A}$, and edge types as $e, e \in \{for, against, vote\}$. The neighbors set of v_i under e is denoted as $\mathcal{N}_{i,e}$, the weight matrix as $\mathbf{W}$, and the attention parameter between v_i and v_j under e is $\alpha_{i,j,e}$. The process of GATv2 is formally described as follows:

$$\begin{aligned} \mathbf{h}^{(l)}_{i,e} &= \sigma\left(\textstyle\sum_{j\in\mathcal{N}^{(l)}_{i,e}} \alpha^{(l)}_{i,j,e} \cdot \mathbf{W}^{(l)}_e \mathbf{x}^{(l)}_j\right), \\ \alpha^{(l)}_{i,j,e} &= \\ &\frac{\exp(a^{(l)T}_e \text{LReLU}(\mathbf{W}^{(l)}_e \cdot [\mathbf{h}^{(l)}_{i,e} \| \mathbf{h}^{(l)}_{j,e}]))}{\sum_{k\in\mathcal{N}^{(l)}_{i,e}} \exp(a^{(l)T}_e \text{LReLU}(\mathbf{W}^{(l)}_e \cdot [\mathbf{h}^{(l)}_{i,e} \| \mathbf{h}^{(l)}_{k,e}]))}, \\ \mathbf{x}^{(l+1)}_i &= aggr(\mathbf{h}^{(l)}_{i,e}), \end{aligned} \tag{1}$$

where l denotes the layer index, and a is a learnable attention weight vector. $\text{LReLU}(\cdot)$ is a LeakyReLU function. The output of the second GATv2 layer is $\mathbf{X}^{(2)}$, which we first concatenate into the vector $\mathbf{X}_1 = [\mathbf{x}^{(2)}_1, \mathbf{x}^{(2)}_2, \mathbf{x}^{(2)}_3, \mathbf{x}^{(2)}_4]$. This is followed by ReLU activation function and a dropout layer (Veličković et al., 2018). The final output $\hat{\mathbf{y}}$ is obtained through a linear layer with a dimensionality of 4 and a softmax layer, i.e., $\hat{\mathbf{y}} \in \mathbb{R}^{1\times 4}$. The ground truth is $\mathbf{y} \in \mathbb{R}^{1\times 4}$. When training the expert model, a cross-entropy loss function is imposed, i.e.,

$$\mathcal{L} = \text{Cross\_Entropy\_Loss}(\mathbf{y}, \hat{\mathbf{y}}). \tag{2}$$

4.2.3 Inference of LLM-driven Judge Agent

With the expert model M_e trained on simulated data, an LLM-driven judge agent gets access to expert observations in real games, even when their initial states differ from those in our simulated games. As illustrated in Figure 5, the data of interaction

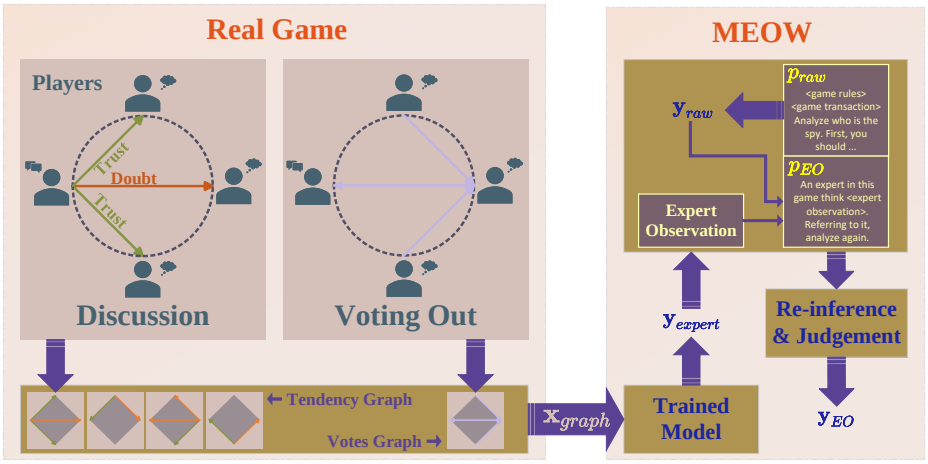

Figure 5: The framework of MEOW in the four-player version of "Find The Spy".

behaviors and players' states recorded in text form are added to the initial prompt, represented as p_{raw}, which allows the judge agent to infer which player is the "spy" for the first time. We denote the inference result as $\mathbf{y}_{raw}$. Concurrently, the real data are transformed into graph-structured data $\mathbf{x}_{graph}$ using the same method as in simulated data and then inputted into the expert model. Following the judge agent's initial inference, the output $\mathbf{y}_{expert}$ of the expert model is transformed into prompts p_{EO}, which instruct the judge agent to make another inference. We anticipate that the weighted average F1-score between the refined inference result $\mathbf{y}_{EO}$ and the ground truth $\mathbf{y}$ is the lowest, i.e.,

$$f(\mathbf{y}_{true}, \mathbf{y}_{pred}) = \text{WA\_F1}(\mathbf{y}_{true}, \mathbf{y}_{pred}),$$
$$f(\mathbf{y}, \mathbf{y}_{EO}), f(\mathbf{y}, \mathbf{y}_{expert}) \geq f(\mathbf{y}, \mathbf{y}_{raw}).$$

5 Experiment

5.1 The Choice of LLM

During the demo design stage of our experiment, internLM-20B (Team, 2023) was as our language model. Despite its superior performance over ChatGPT on major evaluation datasets such as RACE (Lai et al., 2017), it still struggled to reason effectively with its common sense obtained through pre-training. The agents driven by it failed to comprehend the rules provided by our prompts and tend to repeat the same answer to the same questions about different players, mirroring their initial responses. Therefore, to balance cost and effectiveness, we opt for the latest ERNIE-Bot-8k (Wang et al., 2021) as the LLM model used in the experiments. Since ERNIE-Bot-8k exhibits superior reasoning ability in Chinese compared to English, we design the prompts in Chinese. The examples in this paper are summarized and translated from Chinese.

5.2 Agent Response

Reviewing the simulation log, we confirm that agents driven by ERNIE-Bot-8k have the ability to understand the game rules and make rational analyses of other players' utterances and behaviors. For example, receiving the word "sunrise", the player agent Alice noticed an anomaly in Daniel's description.

```
User: In this round, the fourth player
Daniel's decription of his word is "light
fades". <prompts instructing Alice to make
analysis based on common sense>
Assistant: <Alice's analysis>
...
(discussion process starts)
...
User: Reviewing utterances in this round,
analyze other player's motivation.
Assistant: <analysis of Bob and Carol>
Daniel describes "light fades", which I
think is more likely the "Spy", since the
sky gradually lights up at sunrise, rather
than darkens.
```

Moreover, still in this game, Bob received the same word as Daniel which means Alice was the "Spy". After Alice doubted Daniel in statement, Bob raised the opposite doubt.

```
Assistant: <thinking Daniel received the
same word as his>
...
(discussion process starts)
...
User: <(broadcast) Alice's statement
(doubting Daniel)>
```

dataset	graphs	nodes	edges
D_1	261	1044	3551
D_2	181	724	3966

Table 1: Summary of the datasets used in our experiments.

```
Assistant: <analysis of other players
(trusting Daniel)> For Alice, I am against
her. Her doubts about Daniel may have been
strategic to increase her credibility. I
think that during the game, players can
be strategic and deceptive in their
actions and speech.
```

These two agent response samples illustrate the feasibility of simulating communication games with generative agents, and the agents are able to reason with knowledge like human players during the simulated game.

5.3 Dataset

We utilize the aforementioned dataset construction method in Section 4.2.1. Under diverse settings of words players receive, the single-round dataset D_1 consists of 261 valid heterogeneous graphs, totaling 1044 nodes and 3551 edges. The two-round dataset D_2 contains 181 valid heterogeneous graphs, amounting to 724 nodes and 3966 edges. To ensure the stability of our experiments, we implement ten different random data split configurations, yielding ten distinct training and test sets that respectively represent simulated games and real games. A summary of the statistical features of these datasets is presented in Table 1.

5.4 Model Training

The architecture of the two models is detailed in Section 4.2.2. Each dataset is systematically partitioned into ten distinct training and test sets, each of which is used to train an expert model. To ensure robustness and generalize the model's performance across diverse data scenarios, a four-fold cross-validation strategy is adopted, and training stops if the model's performance on the validation set consistently deteriorates over a continuous span of K epochs, compared to the best validation loss identified previously.

Denoting H_i, O_i, $i \in \{1, 2\}$ as the numbers of attention heads, output channels of the ith GATv2 layer, α as learning rate, λ as weight decay rate, p as dropout rate, the hyperparameters $(H_1, O_1, H_2, O_2, \alpha, \lambda, p, K)$ of the first and second expert model are respectively $(6, 32, 6, 16, sum, 0.0001, 0.0005, 0.5, 30)$ and $(6, 32, 6, 18, sum, 0.0001, 0.0005, 0.5, 50)$.

Method	Round	Acc.	WA-F1
EB w/ CoT	1	30.66	29.79
Expert	1	31.60	31.36
EB w/ CoT & EO	1	30.43	30.59
EB w/ CoT	2	28.72	28.96
Expert	2	38.85	38.01
EB w/ CoT & EO	2	36.82	36.50

Table 2: Ablation studies conducted on MEOW. 'Acc.' denotes the accuracy score, and 'WA-F1' represents the weighted average F1-score. In the results generated by EB, responses such as "I'm not certain" are considered random predictions, with their randomness determined based on multi-turn dialogue logs.

5.5 Ablation Study

To validate the effectiveness of MEOW, we perform ablation studies on three distinct inference approaches.

- **ERNIE-Bot with Chain-of-thoughts (EB w/ CoT):** Provide game rules, hints of words, and game broadcast messages including descriptions, utterances, and votes. Then, the judge agent is prompted to infer the agents' actions step by step, following a predefined sequence of steps that includes a strategy review, utterances analysis, and motivation inference, culminating in a final identity judgment.
- **Expert:** The output of the expert model, trained on simulated data, i.e., the expert observation, is considered as the final identity judgment.
- **ERNIE-Bot with Chain-of-thoughts and Expert Observation (EB w/ CoT & EO):** Following the same process as EB w/ CoT, the judge agent is prompted to re-infer the game situation and determine whether to adjust the final judgment based on the expert observation.

For the purpose of avoiding the advantage of "Expert" on training sets, ablation studies are exclusively conducted on the test sets. We perform ten independent experiments with distinct randomized training and test sets to guarantee the reliability of our results. The average statistic of the eight sets,

Method	Round 1	Round 2
EB w/ CoT	0.0419	0.0635
Expert	0.0298	0.0232
EB w/ CoT & EO	0.0146	0.0288

Table 3: Standard deviation of Weighted-average F1 score in round 1 and round 2.

representing the median performance among the ten sets, is presented in Table 2.

In the first round, "EB w/ CoT" attains an accuracy score of 30.66 and a WA-F1 score of 29.79. "Expert" outperforms by approximately 1 percent, achieving an accuracy score of 31.60 and a WA-F1 score of 31.36. The combined method "EB w/ CoT & EO" registers an accuracy score of 30.43 and a WA-F1 score of 30.59, approximating the average of "EB w/ CoT" and "Expert". Similarly, in the second round, "EB w/ CoT" keeps a better performance than "Expert", and "EB w/ CoT & EO" remains intermediate to the two methods. In both rounds, the Expert model exhibits superior performance compared to the CoT methods, and expert observation effectively aids in correcting some errors in the initial inference, as reflected in the statistics.

Besides, as is shown in Table 3, after refinement of expert observations, the standard deviation of WA-F1 score reduces significantly. This indicates that MEOW can enhance the stability of LLM agents' reasoning while improving its performance.

6 Discussion

Through our experiments, training an expert model based on simulated data is both feasible and beneficial for addressing problems in complex human systems. In the absence of any conversation information in a new game, expert models outperform manual CoT methods. On one hand, simulation expands the available data of states in a system, enabling models trained on these data to "experience" more, thereby refining LLM inference even without any historical data. On the other hand, in complex human systems, current LLMs with CoT prompts are not capable of fully comprehending human behaviors and other system forms. As a result, in our two-round experiments, "LLM w/ CoT" fails to recognize that the player voted out in the first round cannot be the "spy", and its performance

```
The results of the first round of voting
are: Alice votes for Carol, Bob votes for
Alice, Carol votes for Bob, and Daniel
votes for Bob. Thus, Bob is eliminated.
...
Next up is player Bob, who I consider to
be the "spy".
```

falls below 33 percent, which is the expectation of random guessing.

When adjusting the parameters of the judge agent, we discover that at the setting of $(top\_p, temperature) = (0.85, 0.8)$, the judge agent achieves an optimal balance between sticking to itself and illogical inferences. In this setting, expert observations do correct some inferences.

```
According to the judgment of game experts,
Daniel is likely to be the "spy". I think
there is some truth to it, because Daniel
did not give any clear words related to
bees in the description and discussion
session, and his descriptor "unique" was
relatively vague, which is consistent with
the characteristics of the "spy" trying
to hide his identity.
```

Therefore, MEOW is never the substitute for FT, CoT, ICL, and RAG, but rather to collaborate with them to enhance the performance of inference in complex human systems. By leveraging these rapidly evolving reasoning methods, the simulation process in MEOW becomes more authentic, thereby increasing the value of expert observation.

7 Conclusion

In this paper, we propose a novel methodology aiming at enhancing the reasoning ability of LLMs in addressing problems concerning complex human systems through simulation. It brings unprecedented human behavior simulation and has the potential for addressing data collection challenges out of privacy, security, etc. in special scenarios. Leveraging the simulated data, we present MEOW, a framework that employs ML models to learn from these data and generate expert observations. By referring to these expert observations, LLMs can refine their inferences, typically yielding results better than those obtained using the CoT-only method. We believe that beyond the realm of communication games, in real-world complex human systems such as sociology and economics, both the generative-agents-based simulation technique and MEOW hold promise as effective methodologies. We are looking forward to the development of more powerful LLMs that can further enhance the effectiveness of our proposed framework.

Limitations

Hallucination is the primary limitation when scaling generative-agents-simulation and subsequent MEOW. Given that LLMs generate text in a "token by token" manner, setting a relatively low temperature to achieve the desired diversity in simulation can lead to errors in some key tokens among all dialogues. For instance, during the discussion process, double quotation marks are followed by identities or word descriptions. If we set the temperature to 0.8, we observe the following response in the simulation,

```
Player Bob's description "Spy" ...
```

which results in illogical reasoning. The security of LLMs presents another limitation if we aim to utilize generative agents and MEOW in human-related security scenarios. To prevent negative generation of LLMs, token chains that violate laws and social norms are penalized during pre-training. However, this restricts the ability of generative agents to simulate offensive attackers. Cost is the final limitation, and its impact becomes more significant when scaling. Implementing generative-agent-based simulation and MEOW requires large volumes of multi-turn dialogues, with most previous turns needing to be retained as context for the next generation. This implies that after completing one generation, there are fees for its tokens in every subsequent turn. In our experiments, using GPT-4 would cost us 5400 dollars, and this is only for a four-player version. As the number of agents increases, there will be more interactions, leading to nonlinear growth in cost.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Shaked Brody, Uri Alon, and Eran Yahav. 2021. How attentive are graph attention networks? In *International Conference on Learning Representations*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Yifeng Cui, Kim B. Olsen, Thomas H. Jordan, Kwangyoon Lee, Jun Zhou, Patrick Small, Daniel Roten, Geoffrey Ely, Dhabaleswar K. Panda, Amit Chourasia, John Levesque, Steven M. Day, and Philip Maechling. 2010. Scalable earthquake simulation on petascale supercomputers. In *SC '10: Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–20.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*.

Larry Eisenberg and Thomas H Noe. 2001. Systemic risk in financial systems. *Management Science*, 47(2):236–249.

Chen Gao, Xiaochong Lan, Zhihong Lu, Jinzhu Mao, Jinghua Piao, Huandong Wang, Depeng Jin, and Yong Li. 2023. S^3: Social-network simulation system with large language model-empowered agents. *arXiv preprint arXiv:2307.14984*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *Sort*, 2(4):0–6.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Shuodi Hui, Huandong Wang, Zhenhua Wang, Xinghao Yang, Zhongjin Liu, Depeng Jin, and Yong Li. 2022. Knowledge enhanced gan for iot traffic generation. In *Proceedings of the ACM Web Conference 2022*, pages 3336–3346.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Edward N Lorenz. 1963. Deterministic nonperiodic flow. *Journal of atmospheric sciences*, 20(2):130–141.

Robert M May. 1976. Simple mathematical models with very complicated dynamics. *Nature*, 261(5560):459–467.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–22.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. 2017. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2107–2116.

InternLM Team. 2023. Internlm: A multilingual language model with progressively enhanced capabilities. https://github.com/InternLM/InternLM.

Robert Tinn, Hao Cheng, Yu Gu, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2023. Fine-tuning large neural language models for biomedical natural language processing. *Patterns*, 4(4).

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. *International Conference on Learning Representations*.

Shuohuan Wang, Yu Sun, Yang Xiang, Zhihua Wu, Siyu Ding, Weibao Gong, Shikun Feng, Junyuan Shang, Yanbin Zhao, Chao Pang, et al. 2021. Ernie 3.0 titan: Exploring larger-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2112.12731*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Liang Zhou, Xuewei Dang, Qinke Sun, and Shaohua Wang. 2020. Multi-scenario simulation of urban land change in shanghai by random forest and ca-markov model. *Sustainable Cities and Society*, 55:102045.

A Game Processes

1. At the beginning of a game, the referee distributes words to all players, and then several processes loop until one side wins. A loop of 2, 3, 4 is called one round, and a game may consist of one or more rounds;

2. Description process: Players take turns to deliver statements, using a word (cannot be their own word or include all their fellow players who used their words before) to describe their received word;

3. Discussion process: After all players give descriptions of received words, they take turns to express opinions on other players' identities.

4. Voting process: Each player chooses a possible "spies" player to vote for him/her. The Player getting the most amounts of votes is eliminated from the game (If several players get the same number of votes, the player whose first vote is got earlier will be eliminated.). The referee will announce which player is eliminated by voting and reveal his/her identity. The eliminated player will be excluded from the rest of the game.

Other rules: a) In each player's utterances at any time it is not allowed to use the word he/she receives. b) In each player's description of words in each round they must not use words used by other players before.

B Six Phases

1. **Game initialization**
   (a) **Game initialization phase:** Every player agent is prompted the game rules, game skills, the players in the game, its received word, and a request to confirm above information.

2. **Description process**
   (a) **Identity inference phase:** Before describing the word, the player agent is prompted to infer its identity in the game based on other players' utterances and behaviors so as to guess the word which players of the opposite identity received.
   (b) **Word description phase:** The player agent is first prompted to recall the word they received and the game rules. Consequently, it establishes its description strategy (be honest or tell a lie). Finally, we prompt the player agent to give its description in textual JSON format so that the game engine can complete the transformation efficiently, and this description is the only information open to other players in this process.
   (c) **Description analysis phase:** After a player agent makes its word description, the rest player agents will be prompted to analyze it. They are requested to judge whether it is proper to describe their received words and further consider its identity.

3. **Discussion process**
   (a) **Statement and discussion phase:** Firstly, a player agent is prompted to recall its identity and other players' descriptions and statements in this round. Secondly, it needs to choose the players it will be for and against, which will have to be embodied in his following statement. Thirdly, we prompt it to confirm its tendency in textual JSON format. Finally, it makes its statement corresponding to its tendency in this round, and this statement is the only information open to other players in this process.

4. **Voting process**
   (a) **Vote phase:** Player agents are prompted the vote rules and have an opportunity to rethink the situation of the current game. Finally, they are requested to make their voting in JSON format, and this is the only information open to other players in this process.