Do LLMs Play Dice? Exploring Probability Distribution Sampling in Large Language Models for Behavioral Simulation

Jia Gu[1,2], **Liang Pang**[1,2*], **Huawei Shen**[1,2], **Xueqi Cheng**[1,2]
[1]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
Correspondence: gujia24@mails.ucas.ac.cn, {pangliang,shenhuawei,cxq}@ict.ac.cn

Abstract

With the rapid advancement of large language models (LLMs) for handling complex language tasks, an increasing number of studies are employing LLMs as agents to emulate the sequential decision-making processes of humans often represented as Markov decision-making processes (MDPs). The actions in MDPs adhere to specific probability distributions and require iterative sampling. This arouses curiosity regarding the capacity of LLM agents to comprehend probability distributions, thereby guiding the agent's behavioral decision-making through probabilistic sampling and generating behavioral sequences. To answer the above question, we divide the problem into two main aspects: sequence simulation with explicit probability distribution and sequence simulation with implicit probability distribution. Our analysis indicates that LLM agents can understand probabilities, but they struggle with probability sampling. Their ability to perform probabilistic sampling can be improved to some extent by integrating coding tools, but this level of sampling precision still makes it difficult to simulate human behavior as agents.

1 Introduction

With the rapid development of artificial intelligence technology, large language models (LLMs), as a key component, have demonstrated powerful language understanding and generation capabilities (Zhao et al., 2023; Kojima et al., 2022). In addition to language generation, people have begun to explore the field of introducing LLMs into the decision-making process, such as simulating human sequential decision-making processes as an agent, and LLM agents have gradually shown their excellent potential in simulating human behaviors and decision-making (Li et al., 2023; Gao et al., 2023; Leng and Yuan, 2024; Brand et al., 2023; Horton, 2023). This brings great convenience to research in fields such as computer science, behavioral science, psychology, and sociology. For example, the core conversational robots (Zhou et al., 2024b) and the social robots of LLMs can more realistically simulate human speech and behavior. LLM agents have shown promising results in various tasks that simulate human behavior (Taylor et al., 2022; Yuan et al., 2022; Aher et al., 2023; He et al., 2024). However, whether LLM agents can effectively simulate human behavior sequences remains an open question and requires careful validation.

Human behavior can often be effectively modeled using Markov decision processes (MDPs). Numerous studies have applied this framework to simulate human behaviors such as planning (Callaway et al., 2017; Guojiang et al., 2007; Redding et al., 2012). In MDPs, sequences of actions follow a specific probability distribution. Similarly, human behavior sequences also adhere to probabilistic distributions (Yan et al., 2012; Hu and Han, 2008; Tenorth et al., 2013). **An LLM-based agent should generate actions that match a probability distribution if it truly simulates human behavior. But can LLM agents do this? To explore this issue, we analyze the probability distribution of action sequences generated by LLM agents.** Language models have been extensively evaluated in various aspects (Liang et al., 2023), and they are capable of probabilistic reasoning (Paruchuri et al., 2024), but there is less research on their understanding and sampling of probability distributions. Therefore, our study focuses on understanding and sampling the probability distribution of actions by LLM agents, without setting up other parts of MDPs in detail.

Figure 1 illustrates our approach to assessing LLM's understanding and sampling capabilities of probability distributions. We outline two tasks of varying complexity: simulating scenarios with explicit probability distributions and generating se-

---

[*] *Corresponding author

Proceedings of the 31st International Conference on Computational Linguistics, pages 5375–5390
January 19–24, 2025. ©2025 Association for Computational Linguistics

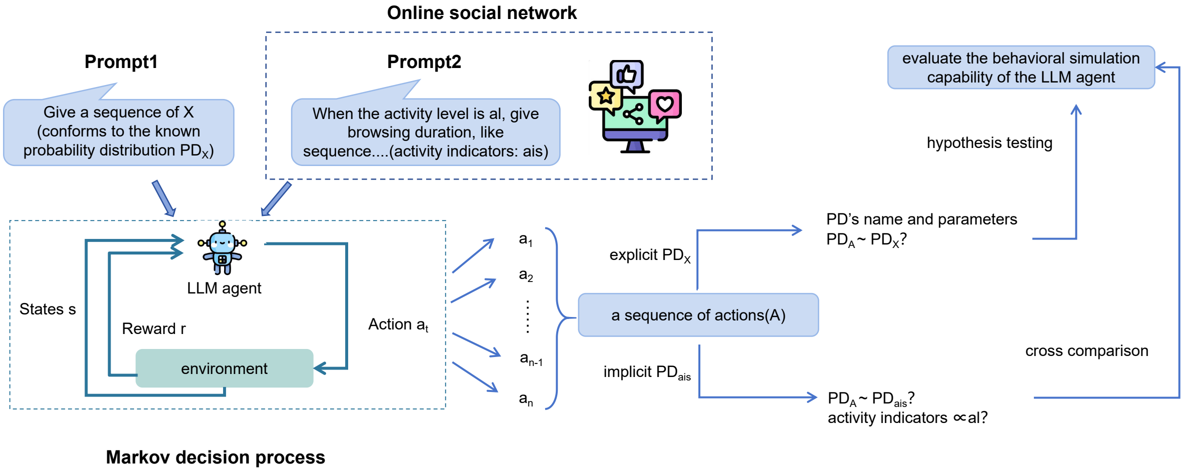


Figure 1: The decision-making process of the LLM agent is a MDP, and the generated action sequence A conforms to a certain probability distribution. We input $Prompt1$ for the explicit probability distribution and $Prompt2$ for the implicit probability distribution, analyze the probability distribution PD_a of A generated by the LLM agent, and finally evaluate the behavioral simulation capability of the LLM agent.

quences with probability distribution is not clear. More specifically, explicit probability distributions usually refer to probability distributions that we can express explicitly, usually described by formulas, parameters, or specific probability density functions (such as the normal distribution). Implicit probability distributions do not have an explicit mathematical representation and are usually defined indirectly through a generative process or model. As shown in the figure 2, in the first scenario, we propose questions about explicit probability distributions. Using these questions to prompt LLM agents for probability sampling, we analyze the content generated by LLM agents. However, obtaining an accurate probability distribution is often challenging in practical scenarios. Therefore, we explore sequence generation in situations where the probability distribution is implicit. Online social networks are increasingly used in real-life scenarios (Sakaki et al., 2013; Panzarasa et al., 2009), so we use these networks as contextual background. By observing varying user activity levels corresponding to different behavior sequences generated by LLM agents, we indirectly assess the rationality of the sequences sampled. Ultimately, we instruct LLM agents to sample from probability distributions by generating code, in order to evaluate whether programming tools can assist LLM agents in improving their sampling success rate.

Our analysis suggests that LLM agents understand probability distributions, but their performance in sampling sequences adhering to probability distributions are limited. However, when coupled with programming tools, LLM agents can achieve higher success rates in sampling explicit distributions by invoking appropriate functions. However, their abilities to sample from complex and implicit probability distributions remain weak. Consequently, prudent consideration is necessary before employing LLM directly to simulate human behavior.

Our main contributions are as follows.

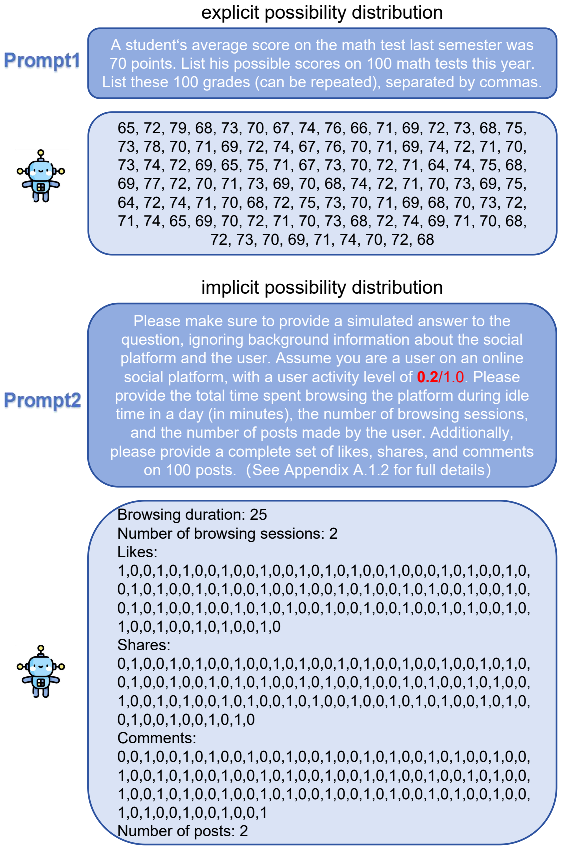
- Explore the ability of the LLM agent to understand and sample probability distributions from both the perspectives of explicit and implicit probability distributions.
- An idea of using programming tools to assist sampling is proposed is effective in explicit probability distributions.
- Our experimental results reveal the limitations of LLM agents in terms of probability distributions when simulating human behavior.

2 Background

2.1 MDPs of Human Behavior

The modeling of human behavior in MDPs could include the following aspects (Tan and Cheng, 2009):

Human behavior can be understood as transitions between different states of state space (S), which may include emotional states, social identities, and more. In each state, humans can choose from a series of possible actions or decisions of action


explicit possibility distribution
Prompt1
implicit possibility distribution
Prompt2

A student's average score on the math test last semester was 70 points. List his possible scores on 100 math tests this year. List these 100 grades (can be repeated), separated by commas.

```
65, 72, 79, 68, 73, 70, 67, 74, 76, 66, 71, 69, 72, 73, 68, 75,
73, 78, 70, 71, 69, 72, 74, 67, 76, 70, 71, 69, 74, 72, 71, 70,
73, 74, 72, 69, 65, 75, 71, 67, 73, 70, 72, 71, 64, 74, 75, 68,
69, 77, 72, 70, 71, 73, 69, 70, 68, 74, 72, 71, 70, 73, 69, 75,
64, 72, 74, 71, 70, 68, 72, 75, 73, 70, 71, 69, 68, 70, 73, 72,
71, 74, 65, 69, 70, 72, 71, 70, 73, 68, 72, 74, 69, 71, 70, 68,
72, 73, 70, 69, 71, 74, 70, 72, 68
```

Please make sure to provide a simulated answer to the question, ignoring background information about the social platform and the user. Assume you are a user on an online social platform, with a user activity level of 0.2/1.0. Please provide the total time spent browsing the platform during idle time in a day (in minutes), the number of browsing sessions, and the number of posts made by the user. Additionally, please provide a complete set of likes, shares, and comments on 100 posts. (See Appendix A.1.2 for full details)

```
Browsing duration: 25
Number of browsing sessions: 2
Likes:
1,0,0,1,0,1,0,0,1,0,0,1,0,0,1,0,1,0,1,0,0,1,0,0,0,1,0,1,0,0,1,0,
0,1,0,1,0,0,1,0,1,0,0,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,0,1,0,
0,1,0,1,0,0,1,0,0,1,0,1,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,1,0,0,1,0,
1,0,0,1,0,0,1,0,1,0,0,1,0
Shares:
0,1,0,0,1,0,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,0,1,0,0,1,0,1,0,
0,1,0,0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,0,
1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,
0,1,0,0,1,0,0,1,0,1,0
Comments:
0,0,1,0,0,1,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,0,
1,0,0,1,0,1,0,0,1,0,0,1,0,1,0,0,1,0,0,1,0,1,0,0,1,0,0,1,0,1,0,0,
1,0,0,1,0,1,0,0,1,0,0,1,0,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,0,
1,0,1,0,0,1,0,0,1,0,0,1
Number of posts: 2
```

Figure 2: Examples of our experiments under a explicit probability distribution and an implicit probability distribution.

space (A). These actions lead to state changes with certain probabilities, represented by the state transition probability P. Human behavior is driven by rewards and punishments, described by the reward function R. To maximize benefits, humans adopt various strategies, captured by a policy π, which maps the probability of taking action a in state s.

Humans have long-term goals like life satisfaction or career achievement, while the objective in an MDP is to find a policy π that maximizes the expected cumulative reward.

Based on the above discussion, human behavior can usually be regarded as conforming to a certain probability distribution. Given characteristics, there is a defined probability distribution (Viti et al., 2015). Some simple behaviors conform to common probability distributions (Yan et al., 2012; Hu and Han, 2008; Wang et al., 2011). Additionally, many real-world human behaviors may be complex and composed of multiple distributions (Tenorth et al., 2013; Sorensen et al., 2017).

Human behavior can be modeled as MDPs, where actions in MDPs can be interpreted as behaviors that follow a series of probability distributions (Korthikanti et al., 2010). If LLM agents can accurately simulate human behavior, the sequences of behaviors they generate should align with these probability distributions.

2.2 Large Language Models as Agent

In recent years, with the rapid development of LLM, more and more studies have discovered the great potential of LLM agents in simulating user-generated behaviors. Behavior simulation can generate user research data and facilitate research in recommendation systems, human-computer interaction, social science research, etc. (Hämäläinen et al., 2023; Kim and Lee, 2023; Li et al., 2023). Many studies have shown that LLM agents can exhibit certain social behavior capabilities, simulate individuals, and imitate human behavior (Gao et al., 2023; Leng and Yuan, 2024).

In addition to the analysis and research on the human-like behavior of LLM agents in various fields, owing to the complexity of human thinking and their outstanding learning and tool usage abilities, several studies (Yao et al., 2023; Schick et al., 2023; Shinn et al., 2023; Zhou et al., 2024a) have designed an agent framework with LLMs as the core. This framework provides LLMs with additional modules such as profile management, reasoning, and tool usage, enabling more accurate simulation of human speech and behavior.

Simulating human behavior is challenging for artificial intelligence due to its complexity, and the capability of LLM agents to do so through probability distribution is still being explored.

3 Verification Methodology

We analyzed the relationship between human behavior and MDPs, focusing on probabilistic characteristics, and decided to study the probability distribution understanding and sampling capabilities of LLM agents. To determine if LLMs can effectively sample actions, we designed experiments to address the following four research questions:

RQ1: Can LLM agents understand probability distributions?

RQ2: Can LLM agents sample simple explicit probability distributions?

RQ3: Can LLM agents sample complex implicit probability distributions?

RQ4: Can LLM agents improve their probability distribution sampling ability by combining programming tools?

To verify these issues, we evaluated LLMs on explicit and implicit behavioral probability distributions and used code tools to explore their probability distribution sampling ability.

3.1 Explicit Probability Distribution

To verify RQ1 and RQ2, we compared the expected probability distribution with the distribution of generated sequences using hypothesis testing.

3.1.1 Experimental Design

Firstly, given a question $prom$ like Prompt1 in Figure 2 containing state s, the sequence of actions x in the question theoretically conforms to an exact probability distribution $PD(x \mid s)$. We ask the LLM agent to predict a sequence of actions a according to the $prom$. Our expectation is as follows.

$$PD(a \mid s) = PD(x \mid s) \tag{1}$$

For instance, based on the average math scores from the last semester, we ask LLM agents to predict the future scores which conform to normal distribution, as illustrated in Figure 2.

$prom_D$
$prom$ + What probability distribution does this sequence conform to

Table 1: $prom_D$ in experiments with explicit probability distributions.

Secondly, leveraging the improvement to LLMs through the chain of thought (CoT) approach, we add "what probability distribution does this sequence conform to" to $prom$, resulting in $prom_D$ as Table 1 shows. This guide the LLM agents to identify the name of probability distribution to further explore the probability distribution understanding and sampling capabilities of LLM agents.

We also extract the type of the probability distribution from the LLM agents' answers to verify if they can understand probability distributions.

3.1.2 Evaluation Metrics

We design evaluation metrics to assess the probability distribution understanding and sampling capabilities of LLM agents.

Firstly, considering the name of probability distribution answered by LLM agents, unanswered and incorrect answers are treated as negative samples. Only correct answers are treated as positive samples. The ratio of positive samples to total samples RP is calculated to evaluate the LLM's understanding of probability distributions.

Secondly, to evaluate the LLMs' performance in sampling the probability distribution PD_X, we employ Kolmogorov-Smirnov test (KS test) (Jr., 1951). The statistic D is used to assess whether two samples come from the same distribution. It compares the empirical distribution functions $F_m(x)$ and $G_n(x)$, from PD_X and PD_A respectively, where the empirical distribution function represents the proportion of samples less than or equal to x.

$$D = \sup_x |F_m(x) - G_n(x)|, \tag{2}$$

where $\sup_x$ denotes the supremum over all possible values of x. Then, we assess the fit between the sample sequence and the target probability distribution using the mean p-value p from the KS test. When the p-value is greater than the significance level α, the test is considered to have passed, and the rate of test pass is denoted as RT.

Additionally, the p is obtained on the p-value after outlier processing. Since the calculated average value is easily affected by outliers, the interquartile range (IQR) outlier detection method was used to detect outliers in the experimental data and replace them with the median. In IQR outlier detection, the interquartile range r is the distance between the upper quartile $Q3$ and the lower quartile $Q1$. The normal value interval is $[Q1 - 1.5r, Q3 + 1.5r]$, and other values are considered outliers.

3.2 Implicit Probability Distribution

To address RQ3, we evaluated the simulation capability of LLM agents using online social networks as the background through comparative analysis.

3.2.1 Experimental Design

In online social network platforms, user activity level l represents the user's level of activity on the platform and affects users' behavior or actions. Compared to low-activity-level users, high-activity-level users frequent social platforms more often, post more comments, browse more posts, and are more likely to like and comment on the posts they browse. We call this rule $\mathcal{R}$. Therefore, even though the probability distribution of these behaviors is unknown, a comparative analysis of the probability distributions of sampled results with different l can help determine whether the user action sequences generated by LLM agents adheres to $\mathcal{R}$.

We set the l as states s in the prompt $prom_l$ like Prompt2 in Figure 2, along with six activity indicators that are directly proportional to the level of activity. These indicators include active behavior indicators and interactive behavior indicators. We use $prom_l$ to prompt LLM agents to generate the sequence of actions a. Our expectation is that $PD(a \mid s)$ follows the same $\mathcal{R}$ as $PD(x \mid s)$.

$$PD(a \mid s), PD(x \mid s) \sim \mathcal{R}, \tag{3}$$

where $PD(a \mid s)$ represents the probability distribution of action sequence a generated by LLM agents under state s, while $PD(x \mid s)$ represents the probability distribution of theoretical action sequence x under state s.

See Appendix A.1.2 for the full prompts used in this experiment. The ability of LLMs to simulate user behavior sequences is analyzed by comparing the generated results across different values of l.

3.2.2 Evaluation Metrics

For active behavior indicators, we compare the results generated by LLM agents. Taking active behavior indicator “browsing duration” as an example, browsing duration increases with activity level.

For interactive behavior indicators, we observed contradictory responses from LLMs. When LLM agents were tasked with generating binary sequences of whether to take action, the proportion of actions taken in actual binary sequences is inconsistent with the proportion reported by LLM agents. Therefore, we adopt a more fine-grained proportion of actions taken in actual generated sequence to represent the frequency of actions. Take interactive behavior indicators “likes” as an example, $S_i = \{b_0, b_1, b_2, \ldots, b_{n_i-2}, b_{n_i-1}, b_{n_i}\}$ represents the like sequence sampled by LLM agents when $l = l_i$. Here, n_i is the length of the sequence, b_j represents the j-th element in the sequence, and the value of b_j is 1 or 0, indicating like or not like, respectively. As l increases, so does the frequency of likes. Therefore,

$$\begin{aligned} c_1/n_1 < c_2/n_2 < c_2/n_3 \\ \text{s.t. } l_1 < l_2 < l_3, \end{aligned} \tag{4}$$

where c_i represents the proportion of 1s in S_i.

For the experimental group that meets the above conditions, the simulation is considered successful. All data combinations corresponding to different activity levels are cross-checked to calculate the simulation success rate RS.

3.3 Methods for Code Experiments

Experiments	$prom_{code}$
explicit distribution	Directly give Python code to solve the following problems: + $prom_D$.
implicit distribution	Directly give Python code to solve the following problems: + $prom_l$

Table 2: $prom_{code}$ in code experiments.

In this section, our goal is to answer RQ4. Considering that LLMs have code generation capabilities (Chen et al., 2021), action sequences can be generated and sampled with the help of Python tools. Therefore, we asked LLM agents to solve the above problems by generating Python code through $prom_{code}$ in Table 2. After running codes, we used the same method to obtain results.

In experiments with explicit distributions, LLM agents are required to generate Python code based solely on $prom_D$. The generated Python code is considered correct if the appropriate probability distribution sampling function is called, and there are no other code errors. In other cases, the code is considered incorrect. Therefore, no further KS test is required. The code accuracy is represented by RC. In the experiment of implicit probability distributions, $prom_{code}$ requires the LLM agent to generate Python code. After running the Python code, the results are analyzed using the same method as in the previous experimental method.

4 Experiment

Based on the four questions we raised in Section 3 and the previous validation methods, in this section, we will introduce our experimental setup and present the experimental results in sequence.

4.1 Experimental Setups

4.1.1 Explicit Probability Distribution

We select the following probability distributions. See the Appendix A.1.1 for $prom$ on each distribution. The **Poisson** distribution describes the number of events occurring within a fixed interval of time or space. The **Uniform** distribution describes a situation where all possible values have equal probability. The **Normal** distribution describes data that are symmetrically distributed around the mean.

Experiments	Metrics	Meanings
Explicit distribution	RP	the rate of agents that can correctly answer the probability distribution
	RT	the rate of the KS test that accepts the hypothesis
	p	the mean p-value of the KS test
Implicit distribution	RS	the simulation success rate
Code experiment	RC	the code simulation accuracy

Table 3: Evaluation metrics in experiments

The **Exponential** distribution describes the time intervals between events in a Poisson process, characterized by a constant event rate. The **Binomial** distribution describes the number of successes in a fixed number of independent trials, each with the same probability of success.

4.1.2 Implicit Probability Distribution

In this part of the experiment, we selected three activity levels: 0.2, 0.5, and 0.8, with the maximum activity level set at 1.0. For the activity indicators, we chose the duration of browsing, the number of visits to the social platform, the number of posts in a day, and the likes, reposts, and comments on 100 posts. The values of these indicators will increase as activity level increases.

4.1.3 Evaluation metrics

In this section, we summarize the evaluation metrics used in the experiment as shown in Table 3.

4.1.4 Models

Five large language models were selected for the experiment of explicit distributions: GPT-4 (Achiam et al., 2023), GPT-3.5 (GPT-3.5), Claude 2.1 (Bai et al., 2022), Llama2 (Touvron et al., 2023), and Vicuna (Chiang et al., 2023). Building on this foundation, four additional LLMs were included for the experiment of implicit distributions: ERNIEBot (ERNIE Bot), ChatGLM (Du et al., 2021), Gemini (Team et al., 2023), and Mixtral (Jiang et al., 2024). Given the complexity of implicit distributions and the diversity among models, we conducted more extensive experiments.

4.1.5 Hyperparameter

The hyperparameter temperature of the LLMs is set to 0.9. The model will consider more possibilities when generating text, rather than just selecting the words with the highest probability. Our experiment hopes that the behavior sequences generated by the LLMs will be more diverse, rather than always the same. We explore whether the diverse sequences follow a unified probability distribution rule.

The significance levels commonly used in the ks test are 0.05 and 0.01, and we use the latter in our experiment. The sample size of our KS test is large (100). When the sample size is large, even small differences may be significant, so a smaller significance level can be used to avoid over-interpreting small differences.

Prompts	**Methods**	**Probability Distribution**				
		Poisson	**Uniform**	**Normal**	**Exponential**	**Binomial**
$prom$	one-step	7.0E-21	7.4E-03	1.6E-03	5.8E-33	1.6E-13
	multi-step	2.1E-41	8.5E-03	9.7E-23	5.8E-33	2.6E-01
$prom_D$	one-step	8.5E-09	3.2E-03	4.0E-03	5.8E-33	6.5E-08
	multi-step	2.1E-41	8.6E-03	3.0E-39	1.6E-13	1.3E-03

Table 4: Comparison of the p of different methods for GPT-3.5 under $prom$ and $prom_D$. The larger the p, the better the model simulation.

4.2 Preliminary experiment

When instructing LLM agents to generate sequences, we compared the multi-step method with the one-step method, as shown in Table 4. In the multi-step method, the behavior sequence is formed by generating one action at a time in the same context, repeating it 100 times. While in the one-step method, the behavior sequence is formed by generating a complete sequence of length 100 at once. During the text generation process by LLMs, the prefix that has already been generated is taken into account. And One-step method generally yields similar or better results compared to multi-step method, except in the case of Binomial distribution. Moreover, the majority of results from both methods did not achieve the level of statistical significance. Therefore, considering both effectiveness and efficiency, we opt for the one-step method.

4.3 Experimental Results

The LLM's probability distribution sampling ability was evaluated in the following four aspects. Examples of LLM's answers in the experiment are shown in Appendix A.2. Experimental results reveal that LLM agents exhibit a remarkable capacity to identify explicit probability distributions from contextual clues. However, their ability to sample from distributions is limited, though it improves with the application of the CoT method or the integration of code tools. Despite these advancements, significant challenges persist, particularly with implicit distributions, where task complexity hampers their ability to generate reasonable outcomes. In addition, we found that the length of sequences generated by LLM agents is usually inconsistent with the instructions, which also indicates that their

Models	Metrics	Probability Distribution				
		Poisson	Uniform	Normal	Exponential	Binomial
GPT-3.5	p	7.0E-21	7.4E-03	1.6E-03	5.8E-33	1.6E-13
	RT	0%	5%	25%	0%	0%
GPT-4	p	**6.5E-08**	4.6E-03	**1.1E-02**	**6.5E-09**	**2.9E-03**
	RT	0%	0%	40%	0%	30%
Claude 2.1	p	5.1E-56	**1.2E-02**	9.3E-03	1.2E-10	2.2E-06
	RT	0%	45%	45%	0%	0%
Llama2	p	3.3E-115	7.8E-17	9.2E-10	0.0	2.5E-05
	RT	0%	0%	10%	5%	0%
Vicuna	pn	1.8E-57	1.6E-204	7.1E-03	1.1E-46	3.8E-04
	RT	0%	5%	50%	0%	15%

Table 5: The p and the RT when the distribution is explicit and the prompt is $prom$. Bold data indicates the model with the largest p on this distribution.

Models	Metrics	Probability Distribution				
		Poisson	Uniform	Normal	Exponential	Binomial
GPT-3.5	p	8.5E-09	3.2E-03	4.0E-03	5.8E-33	6.5E-08
	RT	0%	0%	50%	0%	20%
GPT-4	p	6.5E-08	3.2E-03	3.4E-03	**6.5E-09**	1.5E-02
	RT	0%	0%	20%	0%	40%
Claude 2.1	p	**6.5E-06**	**1.1E-02**	**5.6E-02**	1.2E-10	2.3E-04
	RT	0%	55%	45%	0%	10%
Llama2	p	8.8E-46	1.0E-08	1.6E-07	7.8E-38	**1.6E-02**
	RT	0%	0%	10%	0%	50%
Vicuna	p	2.2E-31	1.6E-09	5.8E-03	2.2E-123	3.9E-10
	RT	0%	0%	25%	0%	10%

Table 6: The p and the RT when the distribution is explicit and the prompt is $prom_D$. Bold data indicates the model with the largest p on this distribution.

simulation capabilities are insufficient.

4.3.1 Understanding of Probability Distributions

If LLM agents can identify the probability distribution PD_X that a random variable conforms to, it reflects LLM agents' ability to understand probability distributions concerning question RQ1.

As depicted in Figure 3, when LLM agents are not required to answer probability distributions, except for the Poisson distribution of GPT-4, and the Exponential distribution of GPT-3.5, GPT-4, and Claude2.1, LLM agents rarely take the initiative to answer the question of what distribution it conforms to. However, this does not imply that they do not know the PD_X. In the $prom_D$ scenario, except for partial distributions of Vicuna and Llama2, the RP of other models is higher, reaching more than 80%. Similar results were observed in the $prom_{code}$ experiment. This demonstrates that LLMs have a certain ability to understand probability distributions and can infer the probability distribution based on the questions, although the RP of Vicuna is lower than other models.

4.3.2 Sampling from A Explicit Probability Distribution

In this section, we answered question RQ2. In experiments where the prompt is $prom$, the Poisson distribution experiment of GPT-4, and the Exponential distribution experiment of GPT-3.5, GPT-4, and Claude2.1, the LLM agents could answer the distribution. Therefore, in these four groups of experiments, we reused the data from the experiment where the prompt is $prom_D$.

As shown in Table 5 and Table 6, from the perspective of RT, most models can achieve the highest test pass rate on the Normal distribution, while the RT on other probability distributions is lower, and in many cases the RT is 0%. Judging from the p, the simulation performance of all models on Normal, Uniform, and Binomial distribution is much higher than on the Exponential and Poisson distribution.

At the same time, during our experiments, we found that the Vicuna model often incorrectly answered questions about other distributions as Normal distributions. We speculate that the reason for this phenomenon may be that there is more data consistent with the Normal distribution in the LLMs training data, while there is less data for Exponential and Poisson distribution.

Additionally, we found that experiments using $prom_D$ result in a larger p value compared to those using $prom$. This suggests that this method can enhance the distribution sampling ability of LLM agents, although the p-value often fails to reach α.

Therefore, we conclude that while LLM agents lack the ability to sample from explicit probability distributions, CoT method can be employed to enhance their performance.

4.3.3 Sampling from An Implicit Probability Distribution

For different activity levels l, the RS is calculated based on various activity indicators to measure the rationality of the behavior simulated by LLM agents when the probability distribution is implicit concerning question RQ3.

As seen in Figure 4, most RS of various models are less than 30%, especially Llama2, ERNIEBot, ChatGLM, and Gemini, with an average RS of only slightly more than 10%. Furthermore, there is significant performance variance among models under different activity indicators, indicating that most of the sampling results of implicit probability distributions by LLM agents do not meet the re-

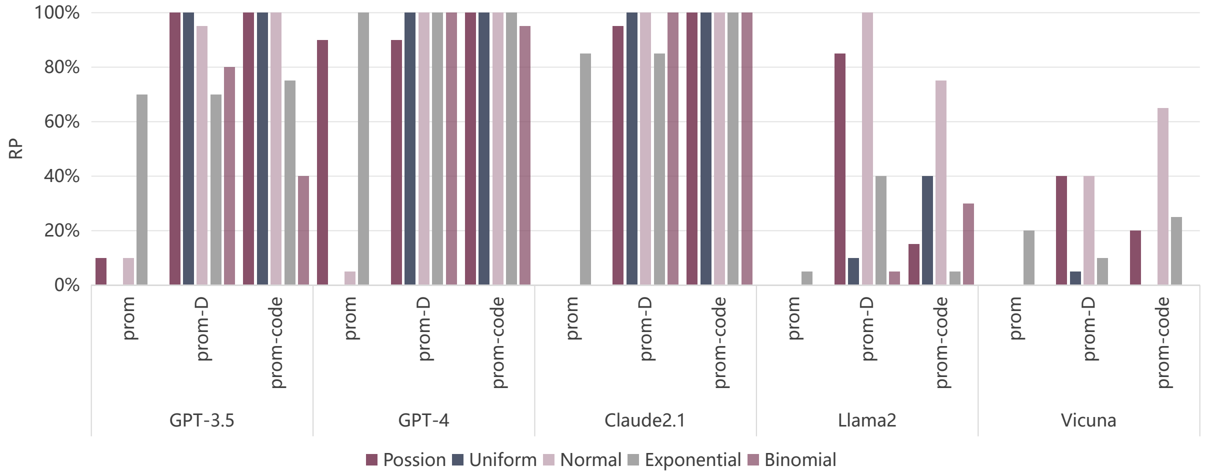


Figure 3: Comparison of the RP of different prompts for each LLM agent under different probability distributions

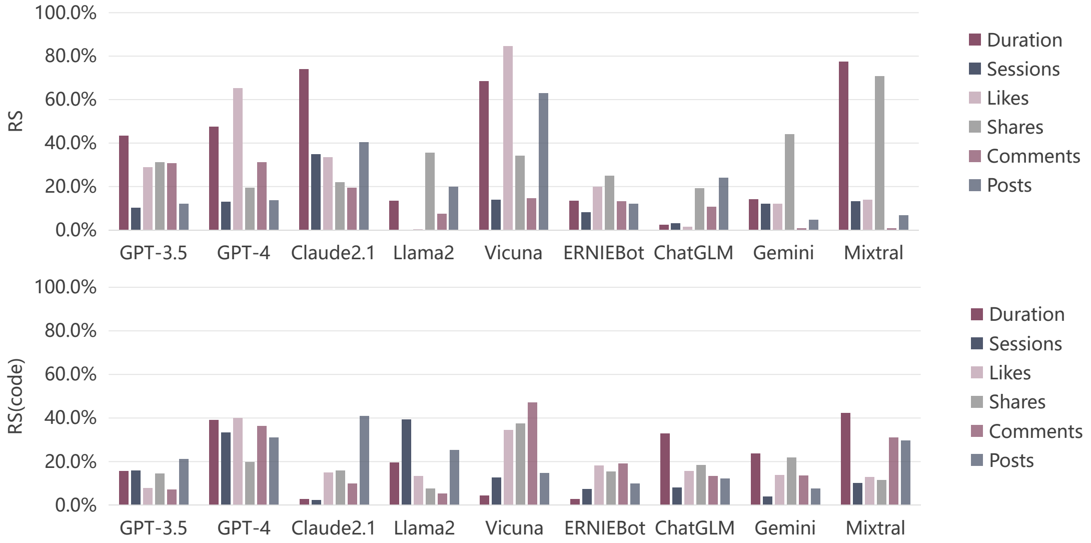


Figure 4: Comparison of the RS of $prom_l$ and $prom_{code}$ for each LLM agent under implicit probability distributions. "Duration" is the browsing duration, "Sessions" is the number of visits to the social platform, "Likes" is the proportion of likes, "Reposts" is the proportion of forwarding, "Comments" is the proportion of comments, and "Posts" is the number of posts.

Models	**Probability Distribution**				
	Poisson	**Uniform**	**Normal**	**Exponential**	**Binomial**
GPT-3.5	100%	100%	100%	75%	100%
GPT-4	100%	100%	100%	100%	100%
Claude 2.1	75%	100%	95%	100%	100%
Llama2	5%	100%	65%	0%	55%
Vicuna	15%	40%	65%	10%	65%

Table 7: In the case of explicit probability distribution, the RC of each model on the probability distribution by generating Python code.

quired conditions, and LLM agents lack the ability to sample implicit probability distributions.

4.3.4 Probability Distribution Sampling Combined with Code Tools

In this section, we answered question RQ4. In the code experiment, $prom_{code}$ leads LLM agents to generate Python code, and the results are obtained by running the Python code.

In experiments with explicit probability distributions, it can be seen from Table 7 and Figure 3 that the RC and the RP of GPT-3.5, GPT-4, and Claude2.1 have reached nearly 100% in most probability distributions, while Llama2 and Vicuna do not perform well. But overall, there has been significant improvement in sampling explicit probability distributions through Python code.

In the experiment with implicit probability distributions, the running results of the code were statistically evaluated in the same way. As shown in Figure 4, we found that combining code tools cannot improve the implicit probability distribution sampling ability of LLM agents, and the RS of each model is still low. Moreover, during the experiment, we found that most LLM agents use code to randomly generate results without considering l. We speculate that because the problem of implicit probability distribution is relatively complex, and there is no suitable Python tool function that can directly generate results, LLM agents still need to generate code through their own reasoning and then obtain results through the code. Unlike the problem of explicit probability distribution, where they only need to reason about the correct probability distribution and parameters, they can usually get the correct code. When reasoning skills are still lacking, it is difficult to generate better code.

5 Conclusion

This paper evaluates the reliability of LLM agents' behavior simulations from the perspective of novel probability distributions. We explore their capabilities through KS tests of explicit distributions and evaluations of implicit distributions. Our findings indicate that while LLM agents can understand of probability distributions, their sampling abilities are insufficient. Consequently, it is challenging to generate behavior sequences that conform to specific distributions solely with LLMs. This limitation may arise from the probabilistic nature of LLMs' word prediction, which results in a layered probability distribution. Even with programming tools, improving sampling performance for implicit distributions remains difficult. Due to the probabilistic nature of actions in MDPs, without effective sampling capabilities, LLM agents struggle to simulate human behavior accurately. Future work will focus on enhancing the LLM agent's probability distribution sampling abilities to improve behavioral simulations.

Limitations

Due to resource limitations, some experiments lack comprehensive evaluation. For example, when comparing the similarity between multi-step generated sequences and sequences containing 100 digits generated at once, only the GPT-3.5 model was evaluated.

Additionally, the experiment in this article is based on prompt engineering and faces issues of instability. Before conducting experiments, we compared various prompts and selected those that were relatively better and elicited better responses from the LLM. Future work will further address this issue by developing more diverse prompts and enhancing the robustness of evaluations.

Ethics Statement

This study adheres to rigorous ethical standards to ensure the integrity and credibility of the research. The primary focus is on evaluating the reliability of LLM agents' behavior simulations, which does not involve any human participants or sensitive data. Transparency is maintained through clear reporting of methods and results. The study was approved by the appropriate ethics review board, and efforts were made to responsibly use resources and consider the future implications of the research findings.

Acknowledgments

This work was supported by the Strategic Priority Research Program of the CAS under Grants No.XDB0680302, the National Natural Science Foundation of China (NSFC) under Grants No.62276248, and the Youth Innovation Promotion Association CAS under Grants No.2023111.

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*.

Gati V Aher, Rosa I. Arriaga, and Adam Tauman Kalai. 2023. Using large language models to simulate multiple humans and replicate human subject studies. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 337–371. PMLR.

Yuntao Bai, Saurav Kadavath, and Sandipan Kundu et.al. 2022. Constitutional ai: Harmlessness from ai feedback. *Preprint*, arXiv:2212.08073.

James Brand, Ayelet Israeli, and Donald Ngwe. 2023. Using GPT for market research. *SSRN Electron. J.*

Frederick Callaway, Falk Lieder, Paul M Krueger, and Thomas L Griffiths. 2017. Mouselab-mdp: A new paradigm for tracing how people plan. In *The 3rd Multidisciplinary Conference on Reinforcement Learning and Decision Making*.

Mark Chen, Jerry Tworek, and Heewoo Jun et.al. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2021. Glm: General language model pretraining with autoregressive blank infilling. *arXiv preprint arXiv:2103.10360*.

ERNIE Bot. 2023. Ernie bot. `https://yiyan.baidu.com/`.

Chen Gao, Xiaochong Lan, Zhihong Lu, Jinzhu Mao, Jinghua Piao, Huandong Wang, Depeng Jin, and Yong Li. 2023. S3: Social-network simulation system with large language model-empowered agents. *Preprint*, arXiv:2307.14984.

GPT-3.5. 2022. Gpt-3.5. `https://platform.openai.com/docs/models/gpt-3-5`.

Wang Guojiang, Wang Zhiliang, Teng Shaodong, Xie Yinggang, and Wang Yujie. 2007. Emotion model of interactive virtual humans on the basis of mdp. *Frontiers of Electrical and Electronic Engineering*, 2(002):156–160.

Perttu Hämäläinen, Mikke Tavast, and Anton Kunnari. 2023. Evaluating large language models in generating synthetic hci research data: a case study. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, New York, NY, USA. Association for Computing Machinery.

Zhiwei He, Tian Liang, Wenxiang Jiao, Zhuosheng Zhang, Yujiu Yang, Rui Wang, Zhaopeng Tu, Shuming Shi, and Xing Wang. 2024. Exploring Human-Like Translation Strategy with Large Language Models. *Transactions of the Association for Computational Linguistics*, 12:229–246.

John J Horton. 2023. Large language models as simulated economic agents: What can we learn from homo silicus? Working Paper 31122, National Bureau of Economic Research.

Hai-Bo Hu and Ding-Yi Han. 2008. Empirical analysis of individual popularity and activity on an online music service system. *Physica A: Statistical Mechanics and its Applications*, 387(23):5916–5921.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *Preprint*, arXiv:2401.04088.

Frank J. Massey Jr. 1951. The kolmogorov-smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78.

Junsol Kim and Byungkyu Lee. 2023. Ai-augmented surveys: Leveraging large language models and surveys for opinion prediction. *Preprint*, arXiv:2305.09620.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213. Curran Associates, Inc.

Vijay Anand Korthikanti, Mahesh Viswanathan, Gul Agha, and YoungMin Kwon. 2010. Reasoning about mdps as transformers of probability distributions. In *2010 Seventh International Conference on the Quantitative Evaluation of Systems*, pages 199–208.

Yan Leng and Yuan Yuan. 2024. Do llm agents exhibit social behavior? *Preprint*, arXiv:2312.15198.

Siyu Li, Jin Yang, and Kui Zhao. 2023. Are you in a masquerade? exploring the behavior and impact of large language model driven social bots in online social networks. *Preprint*, arXiv:2307.10337.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. 2023. Holistic evaluation of language models. *Preprint*, arXiv:2211.09110.

Pietro Panzarasa, Tore Opsahl, and Kathleen M. Carley. 2009. Patterns and dynamics of users' behavior and interaction: Network analysis of an online community. *Journal of the American Society for Information Science and Technology*, 60(5):911–932.

Akshay Paruchuri, Jake Garrison, Shun Liao, John Hernandez, Jacob E. Sunshine, Tim Althoff, Xin Liu, and Daniel McDuff. 2024. What are the odds? language models are capable of probabilistic reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages

11712–11733. Association for Computational Linguistics.

J. D. Redding, Kemal N Ure, J. P. How, M. A. Vavrina, and J. Vian. 2012. Scalable, mdp-based planning for multiple, cooperating agents. In *American Control Conference*.

Takeshi Sakaki, Yutaka Matsuo, Satoshi Kurihara, Fujio Toriumi, Kosuke Shinoda, Itsuki Noda, Koki Uchiyama, and Kazuhiro Kazama. 2013. The possibility of social media analysis for disaster management. In *2013 IEEE Region 10 Humanitarian Technology Conference*, pages 238–243.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, pages 68539–68551. Curran Associates, Inc.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pages 8634–8652. Curran Associates, Inc.

Herb Sorensen, Svetlana Bogomolova, Katherine Anderson, Giang Trinh, Anne Sharp, Rachel Kennedy, Bill Page, and Malcolm Wright. 2017. Fundamental patterns of in-store shopper behavior. *Journal of Retailing and Consumer Services*, 37:182–194.

Chek Tien Tan and Ho-Lun Cheng. 2009. Implant: An integrated mdp and pomdp learning agent for adaptive games. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *Preprint*, arXiv:2211.09085.

Gemini Team, Rohan Anil, and Sebastian Borgeaud et.al. 2023. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Moritz Tenorth, Fernando De la Torre, and Michael Beetz. 2013. Learning probability distributions over partially-ordered human everyday activities. In *2013 IEEE International Conference on Robotics and Automation*, pages 4539–4544.

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*.

Andrea Viti, Alberto Terzi, and Luca Bertolaccini. 2015. A practical overview on probability distributions. *J. Thorac. Dis.*, 7(3):E7–E10.

Peng Wang, Xiao-Yi Xie, Chi Ho Yeung, and Bing-Hong Wang. 2011. Heterogenous scaling in the inter-event time of on-line bookmarking. *Physica A: Statistical Mechanics and its Applications*, 390(12):2395–2400.

Qiang Yan, Lanli Yi, and Lianren Wu. 2012. Human dynamic model co-driven by interest and social identity in the microblog community. *Physica A: Statistical Mechanics and its Applications*, 391(4):1540–1545.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. *Preprint*, arXiv:2210.03629.

Ann Yuan, Andy Coenen, Emily Reif, and Daphne Ippolito. 2022. Wordcraft: Story writing with large language models. In *27th International Conference on Intelligent User Interfaces*, IUI '22, page 841–852, New York, NY, USA. Association for Computing Machinery.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *Preprint*, arXiv:2303.18223.

Junkai Zhou, Liang Pang, Ya Jing, Jia Gu, Huawei Shen, and Xueqi Cheng. 2024a. Knowledge boundary and persona dynamic shape a better social media agent. *Preprint*, arXiv:2403.19275.

Junkai Zhou, Liang Pang, Huawei Shen, and Xueqi Cheng. 2024b. Think before you speak: Cultivating communication skills of large language models via inner monologue. *Preprint*, arXiv:2311.07445.

A Appendix

A.1 Prompts

We guide LLM agents to generate sequences through prompts. The prompts used in the experiment are as follows.

A.1.1 Prompts for Explicit Probability Distributions

This section shows the prompts for explicit probability distributions. Table 8 is $prom$ and Table 1 is $prom_D$.

A.1.2 Prompts for Implicit Probability Distributions

This section shows the prompts for implicit probability distributions. Table 9 is the template of $prom_I$.

Probability distributions	$prom$
Poisson distribution	In the past, an average of 3 babies were born per hour. Estimate the number of babies born per hour 100 times in the future. List these 100 numbers (duplicates are allowed), separated by commas.
Uniform distribution	Simulate 100 rolls of the dice and list the 100 points separated by commas.
Normal distribution	A classmate's average score on the math test last semester was 70 points. List his possible scores on 100 math tests this year. List these 100 grades (can be repeated), separated by commas.
Exponential distribution	Assume that a coffee shop has an average of one customer coming in every ten minutes, guess how long it will take to wait for the next customer 100 times, and list these 100 numbers, separated by commas.
Binomial distribution	Assume you have a 30% probability of answering "0" and a 70% probability of answering "1". Please answer 100 times. All answers are separated by commas.

Table 8: $prom$ for different probability distributions in experiments with explicit probability distributions.

A.2 Experimental Examples

A.2.1 Examples in Explicit Probability Distributions Experiments

Some examples of LLM agents' answers in explicit probability distribution experiments are shown in Figure 5.

A.2.2 Examples in Implicit Probability Distributions Experiments

An example of LLM agents' answer in the implicit probability distribution experiments is shown in Table 10.

A.2.3 Examples in Code Experiments

Some examples of LLM agents' answers in the code experiment are shown in Table 11 and Table 12.

$prom_l$

Please make sure to provide a simulated answer to the question, ignoring background information about the social platform and the user. Assume you are a user on an online social platform, with a user activity level of [l]. Please provide the total time spent browsing the platform during idle time in a day (in minutes), the number of browsing sessions, and the number of posts made by the user. Additionally, please provide a complete set of likes, shares, and comments on 100 posts (using 1 to represent likes, shares, and comments, and 0 to represent no likes, shares, and comments, separated by commas).
Please strictly adhere to the following format for output:
Browsing Duration:
Number of Browsing Sessions:
Likes:
Shares:
Comments:
Number of Posts:

Below is an example of an answer:
Browsing Duration: 30
Number of browsing Sessions: 3
Likes: (a string of 100 0s and 1s, where 1 represents a like and 0 represents no like, separated by commas)
Shares: (a string of 100 0s and 1s, where 1 represents a share and 0 represents no share, separated by commas)
Comments: (a string of 100 0s and 1s, where 1 represents a comment and 0 represents no comment, separated by commas)
Number of Posts: 3

Table 9: $prom_l$ in implicit probability distribution experiments, where l in [] represents different levels of activity.

An example answer of Claude2.1

```
Browsing Duration: 72
Number of Browsing Sessions: 6
Likes: 1,1,1,1,1,1,0,0,1,1,1,0,0,1,1,1,0,0,1,0,1,0,1,1,1,1,0,1,1,0,0,0,
1,0,0,0,0,0,1,0,0,0,1,1,0,1,1,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,1,1,1,0,0,0,
0,0,0,1,1,1,0,1,0,0,0,0,0,1,0,0,1,0,0,0,0,1,1,0,1,1,1,1,0,0,0,0,0,0,0,1,
0,1,0,1,1,1,0,1,0,0,0,0
Shares: 1,1,0,0,0,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,1,0,1,0,0,0,0,0,
1,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,1,0,0,1,0,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,
1,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0
Comments: 0,1,0,0,0,1,1,1,0,1,0,0,0,0,1,0,0,0,0,1,0,1,1,0,0,0,0,0,1,0,1,
1,1,1,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,
1,0,0,1,0,1,0,1,0,0,0,0,0,1,1,1,0,0,1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,
1,0,0,1,1,0,0,0,1,1,1,0,0,1,0,0,0
Number of Posts: 6
```

Table 10: An example answer of Claude2.1 in experiments when the probability distribution is unknown.

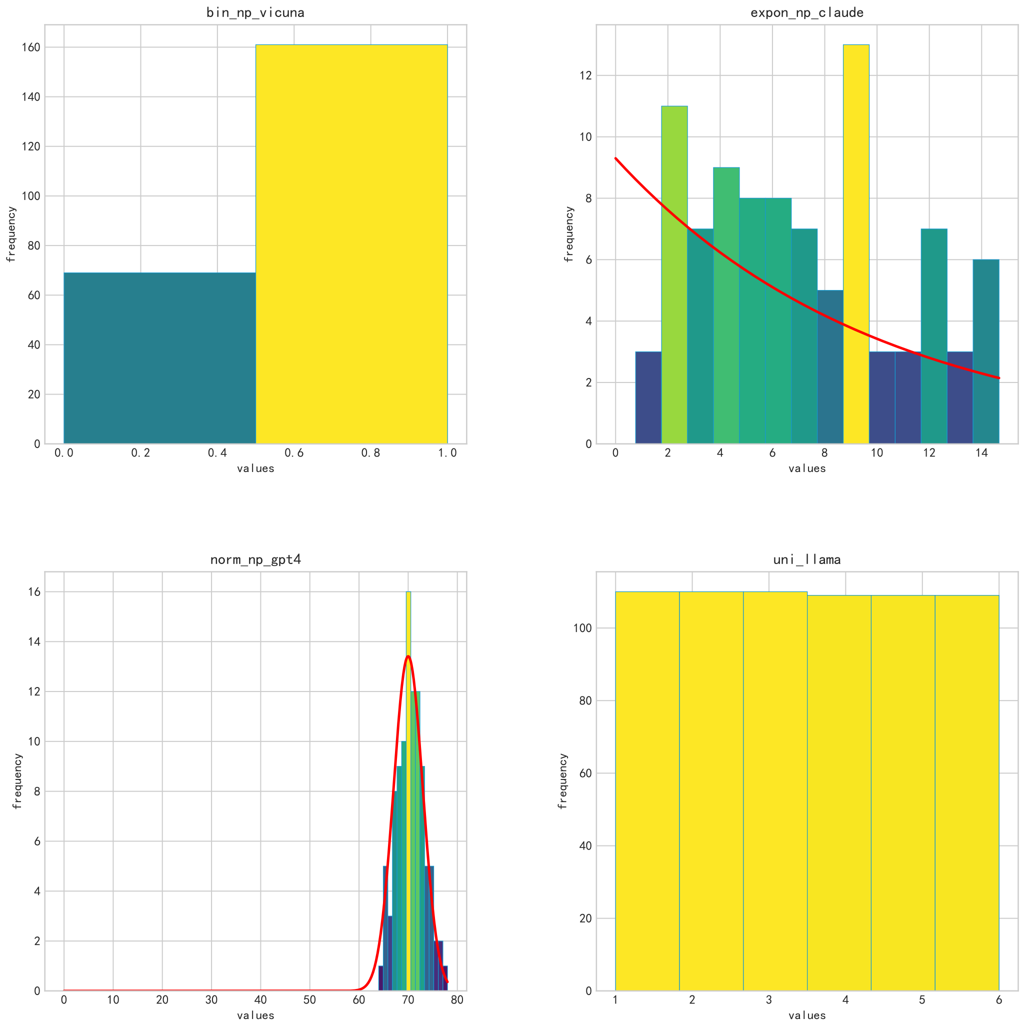


Figure 5: Some examples in explicit probability distributions experiments.The four pictures respectively represent the frequency histograms of vicuna in the Binomial distribution, Claude in the Exponential distribution, GPT-4 in the Normal distribution, and llama in the Uniform distribution under $prom$. The red line represents the ideal probability density function curve.

An example answer of GPT-3.5

```
import random

def next_customer_wait_time():
    # Average time between customers in minutes
    average_time_between_customers = 10

    # Generate random wait time following exponential distribution
    wait_time = random.expovariate(1 / average_time_between_customers)
    return wait_time

# Simulate waiting time for the next customer 100 times
waiting_times = [next_customer_wait_time() for _ in range(100)]

# Print the waiting times separated by commas
print(', '.join(map(str, waiting_times)))
```

Table 11: An example answer of GPT-3.5 in code experiments when the probability distribution is Exponential distribution.

An example answer of GPT-4

```
import random

# Constants
TIME_SPENT_BASELINE \text{=} 45
# Assume an average user spends 45 minutes a day
ACTIVITY_LEVEL = 0.2
NUM_POSTS = 100

# User-specific calculations
time_spent_browsing = TIME_SPENT_BASELINE * ACTIVITY_LEVEL
number_of_sessions = random.randint(1, 5)
# Assume 1-5 sessions for a low activity user
number_of_posts = round(ACTIVITY_LEVEL * 2)
# Assume an active user posts twice per day

# Social interactions are random according to the activity level
likes = [random.choice([0, 1]) for _ in range(NUM_POSTS)]
shares = [random.choice([0, 1]) for _ in range(NUM_POSTS)]
comments = [random.choice([0, 1]) for _ in range(NUM_POSTS)]

# Output
print(f”Browsing Duration: {time_spent_browsing:.2f}”)
print(f”Number of Browsing Sessions: {number_of_sessions}”)
print('Likes:', ':'.join(map(str, likes)))
print('Shares:', ':'.join(map(str, shares)))
print('Comments:', ':'.join(map(str, comments)))
print(f”Number of Posts: {number_of_posts}”)
```

Table 12: An example answer of GPT-4 in code experiments when the probability distribution is unknown.