

Language Models Might Not Understand You: Evaluating Theory of Mind via Story Prompting

Anonymous ACL submission

Abstract

We introduce StorySim, a programmable framework for synthetically generating stories to evaluate the *theory of mind* (ToM) and *world modeling* (WM) capabilities of large language models (LLMs). Unlike prior benchmarks that may suffer from contamination in pretraining data, or rely on an LLM for generation, StorySim produces novel, compositional story prompts anchored by a highly controllable STORYBOARD, enabling precise manipulation of character perspectives and events. We use this framework to design first- and second-order ToM tasks alongside WM tasks that control for the ability to track and model mental states. Our experiments across a suite of state-of-the-art LLMs reveal that most models perform better on WM tasks than ToM tasks, and that models tend to perform better reasoning with humans compared to inanimate objects. Additionally, our framework enabled us to find evidence of heuristic behavior such as recency bias and an over-reliance on earlier events in the story. All code for generating data and evaluations is freely available.¹

1 Introduction

Large language models (LLMs), especially models meant for reasoning, have shown increasing improvement on benchmarks that were once infamously challenging. Contemporary models have made progress in reasoning through math problems, coding, and pure logical deduction (Jaech et al., 2024; Guo et al., 2025; Xia et al., 2025). However, as the field shifts towards improving reasoning abilities, we look to LLM’s abilities to reason about mental states and the perspectives of other intelligent agents, a concept known as *theory of mind* (ToM). This has become increasingly important, as an LLM without a sound ToM might not be able to

understand the perspective of a user with harmful intent, or consider the effect its decisions could have on other people, who have their own goals and wishes. Since reasoning about perspectives requires modeling the surrounding world, we view ToM as an extension of world modeling (WM)—an LLM’s capacity to internally represent world states. Agents with strong WM should exhibit ToM, since other minds exist within the same world.

ToM is an ability that is measurable in children, where we test their ability to understand that other people have their own mental representations of the world that differs from their own. The prototypical ToM test is known as the *Sally-Anne* test (Baron-Cohen et al., 1985), where children are given a story akin to the following:

Sally and Anne walk into a bedroom. Sally places a toy on top of the bed and leaves. Anne takes the toy and hides it under the bed. When Sally comes back into the room, where is the first place she will look for the toy?

It may seem obvious to adults that the first place Sally will look is on top of the bed. However, it has been shown that children under the age of 4.5 think that Sally will look under the bed (Memisevic et al., 2018). This is hypothesized to be due to the fact that they have not yet developed ToM, so they cannot understand the fact that Sally and Anne have a different understanding of the toy’s location.

LLMs have been evaluated on problems of this nature, and initially showed success (Kosinski, 2023). However, small perturbations of this problem often confounds LLMs, indicating that this test has contaminated pretraining data, and LLMs may have overfit to this specific kind of problem, and therefore, they may not necessarily have a sound ToM mechanism (Ullman, 2023). In order to measure ToM ability in a way that is robust against data contamination, we have developed a flexible synthetic benchmark.

¹Code for and data generated by StorySim is available here: <https://anonymous.4open.science/r/StorySim-3A48>

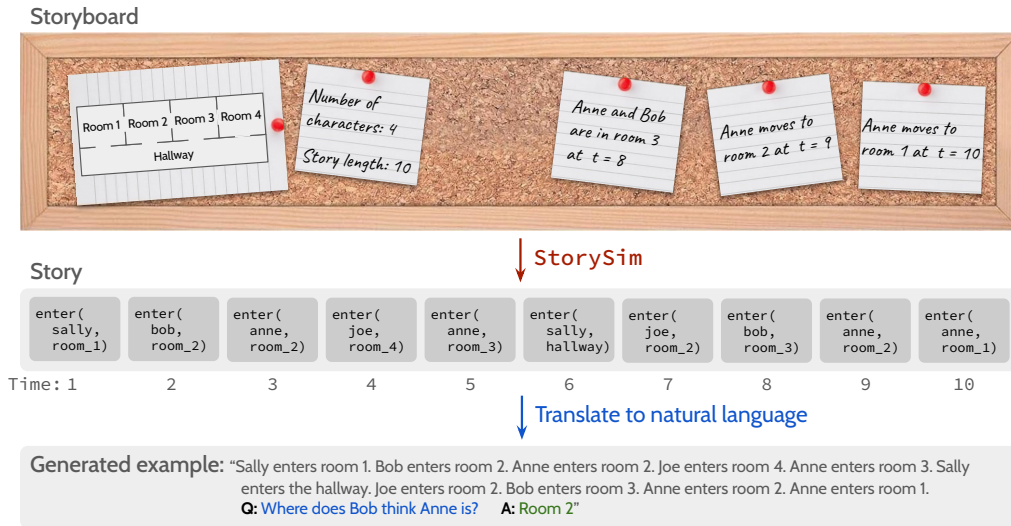


Figure 1: Example STORYBOARD a graph specifying the story setting layout, requested story length, as well as a set of key events with timestamps. StorySim generates a story adhering to the graph and satisfying the required events. In this example, each character can only move to locations corresponding to adjacent vertices in the graph (all characters start in the hallway). In our experiments, each location has possible other rooms to move to. The output of StorySim is translated into a natural language QA example.

We propose StorySim, a framework that allows us to generate stories to evaluate ToM and WM abilities of LLMs. The core events of the story are described by a high level specification called the STORYBOARD. Aside from the STORYBOARD, the rest of the story is randomly-generated, but guaranteed to be cohesive and consistent. An example of this framework is shown in Figure 1. The STORYBOARD provides a description of the setting in which the story will take place, including which locations are accessible from each other, as well as required actions characters need to make, the length of the story, the number of characters, etc. StorySim then takes this and generates a story that is coherent with the STORYBOARD, filling in all other details randomly in a way that is consistent with the provided constraints. Due to the exact specificity of the events that occur in StorySim via the STORYBOARD, the randomness of the events in between, and the length of the stories we generate, the chance that any synthetically-generated story in our experiments has appeared in the training data of any LLMs is astronomically small.

By leveraging StorySim to systematically evaluate current LLMs, we are able to address the following research questions in this paper:

- RQ1** Can we measure ToM abilities of LLMs in a way that is robust to data contamination and heuristics?
- RQ2** To what extent do LLMs rely on memorization and heuristics to answer ToM questions?

RQ3 Can ToM abilities in LLMs be evaluated independently of their general world modeling capabilities? Relatedly, do LLMs show stronger performance on general world modeling tasks or on ToM-specific reasoning?

2 Related Work

Theory of Mind: Existing research benchmarking ToM in language models focuses on prompting with stories followed by question-answering (Grant et al., 2017; Sap et al., 2019), including the TOMI dataset (Le et al., 2019). However, these are outdated, and have very likely leaked into the pretraining data of contemporary LLMs. In fact, Ullman (2023) demonstrates that trivial alterations of the TOMI dataset causes model performance to drop (e.g., Sally-Anne setup in Section 1, but the bed is transparent). In order to address data leakage, newer benchmarks have been created (He et al., 2023; Chen et al., 2024b). Specifically, we position our work as similar to Sclar et al. (2024), where the authors present EXPLORETOM, a framework that allows for graph-based synthetic story generation and question answering. As in our work, this ensures that generated examples are relatively unseen by LLMs. However, while EXPLORETOM uses an LLM to turn fixed contexts into stories, our framework generates sentences directly from a STORYBOARD. This gives precise control over when events occur, enabling targeted tests (e.g., shifting exact state-change locations to probe LLM

behavior). Figure 8 shows that LLM-generated stories often fail to meet such constraints, while StorySim reliably produces stories of any specified length, set of characters, and events. Huang et al. (2024) introduce a complexity measure for ToM tasks and a prompting method that makes implicit character state attributes explicit, offering a formal representation of ToM events similar to ours. However, we test whether LLMs can reason over implicit character states without such scaffolding. Our framework can also add extraneous state changes from irrelevant characters, but, as shown in Figure 6, these have little effect on ToM performance, contrasting with the findings found in previous work.

World Modeling: Though there is much work evaluating aspects of WM abilities of LLMs (Kočiský et al., 2018; Qiu et al., 2025; Qian et al., 2025), there are none, to the best of our knowledge, that examine the relationship between WM and ToM ability. We consider ToM to be an extension of world modeling since ToM requires the explicit modeling of mental states and how they may change over time, and mental states are dependent on the goals, limitations, and perspectives of agents. In the stories we generate, the character’s perspectives only consist of what they observe around them, so to reason over this requires understanding the world in which they live and the limitations of each character’s perception and knowledge. Wang et al. (2024) train a narrator and a role player model to interact within role-playing game environments. This setup implicitly requires both models to construct and maintain an understanding of the environment (WM), while the narrator must also infer the role player’s perspective to guide narrative choices (ToM). However, the task itself does not directly evaluate either ToM or world modeling capabilities in isolation.

Synthetic Datasets: Though existing synthetic datasets meant for benchmarking LLMs exist (Patel et al., 2024; Shvets, 2025; Sood, 2024; Mishra and Prabakeran, 2025), there are none that evaluate ToM and WM abilities as we do. Synthetic datasets have been shown to be an effective method for training LLMs to perform well on logic tasks (Xie et al., 2025), clinical text classification (Tang et al., 2023), deductive reasoning (Saparov et al., 2023), and more. The diversity of synthetic data is an important factor in the evaluation of performance on both in- and out-of-distribution data (Chen et al., 2024a; Havrilla et al., 2024). For this reason, we

developed StorySim to be highly programmable and able to generate a wide range of stories.

3 Methods

Broadly, the StorySim framework works by generating each story in two steps: (1) we create a STORYBOARD, which contains high-level information about the story we want to generate, and (2) using this STORYBOARD, StorySim generates a *story* that meets the specifications of the STORYBOARD and randomly generates sequences of events that fill in the gaps coherently and consistently.

In this dataset, we define a STORYBOARD as $D = (C, \phi, G, E, n)$ where C is a set of characters, G is a directed graph describing locations in which the story happens, E is a programmed set of events that occur in the story, and n is the length of the story in terms of the number of events. The StorySim framework then generates a sequence of n events, where each event can be written in the form $\phi(c, l)$ where ϕ is the action, $c \in C$ is the character performing the action, $l \in V(G)$ is the location of the action, and $V(G)$ are the vertices of G . These locations of the events follow the structure of G , so characters can only move between adjacent vertices in G . The only parts of the story that are deterministic are the events specified in E , which dictates specific time steps at which actions occur. Aside from this, all other events are randomly-generated, but are guaranteed to be cohesive with respect to the events specified in E . An example of such a story is provided in Figure 1. This sequence of events can be simply translated into natural language using templates, which maps each $\phi(c, l)$ into plain text.

Though a simple framework, StorySim can be used to create a wide variety of stories. StorySim does not use template-based generation when creating new stories, and can be used to simulate any story where each action can be modeled as a relation (or set of relations) that act over a subject and node from the graph. The subject does not necessarily need to be a human, and the nodes need not be locations. Additionally, the stories can feature manually-created actions as well, which can add another layer of complexity to the events described by StorySim. See Figure 11 for an example of a more narrative-like story generated by StorySim. Additionally, Figure 14 demonstrates how StorySim can be used to generate stories to test goal-directed and counterfactual reasoning ability; however, we

243 focus on false-belief tasks, as they provide a more
244 controlled test of ToM.

245 While the generated stories are not the most com-
246 plex in terms of prose or grammatical complexity,
247 we can easily add different factors to stories that
248 allow us to evaluate LLMs in a targeted manner.
249 By using events described in simple sentences, we
250 minimize confounding factors related to sentence
251 interpretation, ensuring that any model errors stem
252 from limitations in the targeted ability. We further
253 explore this in Appendix D.1.

254 In our experiments, we create many such STORY-
255 BOARDS that allow us to create stories with ques-
256 tions that can only be answered by models with
257 sufficient ToM and WM abilities. The exact pa-
258 rameters of D in our experiments are specified in
259 Appendix A. We generate stories involving char-
260 acters moving between a set of rooms connected
261 by a hallway. We conducted experiments using
262 different types of settings and found no significant
263 difference in performance, as shown in Figure 6.

264 In order to test ToM abilities, we generate two
265 types of ToM problems:

266 *First-order ToM problem:* Ask about one char-
267 acter’s knowledge of another character’s
268 location. E.g.: **“Where does Bob think
269 Alice is?”**

270 *Second-order ToM problem:* Ask about one char-
271 acter’s knowledge, of another character’s
272 knowledge, of a *third* character’s location.
273 E.g.: **“Where does Bob think Charlie
274 thinks Alice is?”**

275 To create stories that allow us to ask questions
276 like the ones above, we create two types of STO-
277 RYBOARDS which specify events for characters S_1 ,
278 S_2 , and T . In the above examples, S_1 is Bob, S_2
279 is Charlie, and T is Alice. We use StorySim to
280 generate stories to test first-order ToM using the
281 following STORYBOARD: S_1 and T move to the
282 same location at some time, T moves to a different
283 location. Then, unbeknownst to S_1 , T moves to a
284 third location, where they remain for the entirety of
285 the story. We also generate the second-order ToM
286 stories using the following STORYBOARD: where
287 S_1 , S_2 , T move to the same location, then S_2 and
288 T later meet again at a different location, then T
289 moves to a different location on their own.

290 *RQ1 & RQ3:* To answer these research ques-
291 tions, we aim to generate stories that allow us to
292 measure ToM and WM abilities in controlled ex-
293 periments, while minimizing the chance that the
294 model can exploit memorization from pre-training

295 data. Using the second-order STORYBOARD as
296 described above, we generate a large number of
297 stories, and prompt the LLMs with the following
298 three questions:

299 ToM Prompt: *Where does S_1 think S_2 thinks T is?*

300 WM-Human Prompt: *When S_1 and S_2 were in the
301 same room as T , where did T go?*

302 WM-Inanimate Prompt: *When S_1 and S_2 were
303 in the same room as T , where was T moved to?*

304 The first question type requires the LLM to un-
305 derstand the perspective of S_1 and their perceived
306 location of S_2 and T , thereby directly testing its
307 ToM capabilities. The second question type is
308 equivalent to the first, but instead of asking the
309 model to consider the perspective of S_1 , we prompt
310 it to algorithmically search through the story to find
311 an event where S_1 ’s perception of the locations of
312 S_2 and T changed. Thus, the second question is a
313 test of world modeling capability, since we do not
314 ask about the mental state of any character, and an-
315 swering the question requires understanding of the
316 events in the story. Finally, the third question type
317 is equivalent to the second, but instead of describ-
318 ing human subjects, the story describes inanimate
319 objects. This question is meant to test whether
320 the LLM’s world modeling ability is sensitive to
321 whether the entities in the world are animate or
322 inanimate, which shows us the difference in ability
323 when the model is considering humans compared
324 to anything else. The results of evaluating a suite
325 of LLMs on these three prompts is shown further
326 below.

327 *RQ2:* One advantage of the design of StorySim
328 is that it allows us to test for specific heuristics
329 we suspect that models are exploiting to answer
330 questions. We take recency bias as an example of
331 a heuristic that LLMs may use when making pre-
332 dictions. The flexibility of StorySim enables us
333 to easily setup an experiment to test this. Given
334 a STORYBOARD for a first-order or second-order
335 ToM problem, we ask whether the number of time
336 steps between the event where S_1 last perceives
337 T ’s location and the event where T makes their
338 third and final move is predictive of model perfor-
339 mance. Note that when T makes their final move
340 in the story, S_1 ’s earlier knowledge of T ’s location
341 becomes inaccurate. But since T ’s last movement
342 is a more recent event in the story, LLMs may uti-
343 lize a heuristic where they predict T ’s last location,
344 despite the fact that S_1 did not see this movement.
345 We refer to this number of time steps between these

two events as the *mislead distance*. Our initial hypothesis was that increasing the mislead distance would increase the model’s accuracy due to recency bias in the prompt.

To test these heuristics, we create STORYBOARDS for both first- and second- order ToM problems, but we vary the mislead distance. We evaluate multiple LLMs on these examples and evaluate their performance.

In addition, we explore whether and to what extent models have overfit to simpler ToM questions by significantly increasing the number of subjects in the stories. We hypothesize that models may have seen question-answering prompts based on stories during pretraining, but it is unlikely they have such examples where the stories contain large amounts of characters. Being able to do so would imply that the models can generalize more robustly to stories with many mental states, and would therefore provide evidence of more robust/generalizable ToM and WM capabilities. We prompt the models using the *ToM Prompt* and *WM-Human Prompt*, except we ask a first-order ToM question. The results of evaluating the performance of multiple LLMs using this story setup and two prompts are shown in Figure 6.

4 Results

| Task | GPT-4 | R1 |
|--------------|-------|------|
| Sally-Anne | 100% | 100% |
| Random Story | 75% | 87% |

Table 1: Accuracy on Sally-Anne vs other ToM tasks. (**top row**) Models are prompted with the exact Sally-Anne setup shown in Section 1, in a story with 5 events. (**bottom row**) Models are given a randomly-generated first-order ToM story of the same length.

4.1 Sally-Anne Is No Longer a Good Test for ToM

To highlight the inadequacy of benchmarks that rely on the Sally-Anne problem to measure ToM, we used StorySim to mirror the Sally-Anne experiment using GPT-4 and R1, using a STORYBOARD that specifies a first-order ToM problem with 5 events matching the experiment exactly. As shown in Table 1, both of these models had perfect scores on this task, which is far from the results of prompting the models using randomly generated stories of the same length.

4.2 Evaluating ToM vs WM Capabilities

We tested a suite of LLMs using the prompts discussed in Section 3. Starting with a STORYBOARD describing a second-order ToM event, we generate 100 stories consisting of 100 events with which we query each model. Figure 2 shows the results of zero-shot prompting as discussed in Section 3 using GPT-4 (Achiam et al., 2023), R1 (Guo et al., 2025), the Llama 3.1 Instruct family, Llama 3.2 Instruct, Llama 3.3 Instruct (Touvron et al., 2024), and Qwen QwQ (Yang et al., 2025). GPT-4, Llama-3.2, and R1 are the only models that perform better on the ToM prompt than on the WM-Human prompt. While the remaining models score higher on WM-Human than on ToM, their raw accuracies on both tasks are generally higher—or at least comparable—to those of Llama 3.2 and GPT-4. An exception is R1, which achieves the highest ToM accuracy overall. In every model except GPT-4, the WM-Human score is higher than WM-Inanimate, indicating that considering human subjects makes LLMs more accurate in tracking changes in location than with inanimate objects. However, the fact that most of the model’s ToM scores are lower than WM scores leads us to believe that LLMs have greater difficulty when tasked to reason over the mental states of characters.

Since increasing model scale allows LLMs to perform better on many other tasks, we endeavor to test whether or not ToM and WM abilities are enhanced with more parameters. To this end, we test three models of varying sizes in the Llama 3.1 family.

In Figure 2, the Llama 3.1 70B model outperforms both the 8B and 405B models on WM tasks, while ToM scores rise with model size. To explore this further, we evaluated the Llama family in a 3-shot setting (Figure 3), revealing a strong correlation between model size and both ToM and WM-Inanimate performance, with the peak being 405B again. These results suggest that, within the Llama 3.1 architecture, there is a clear correlation between ToM and model size. In the 3-shot setting, this correlation is apparent in WM abilities as well.

4.3 Do LLMs Rely on Heuristics for ToM?

Our next experiments aim to address whether or not LLMs are exploiting simple heuristics to answer our questions rather than relying on an actual ToM mechanism. We used StorySim to create

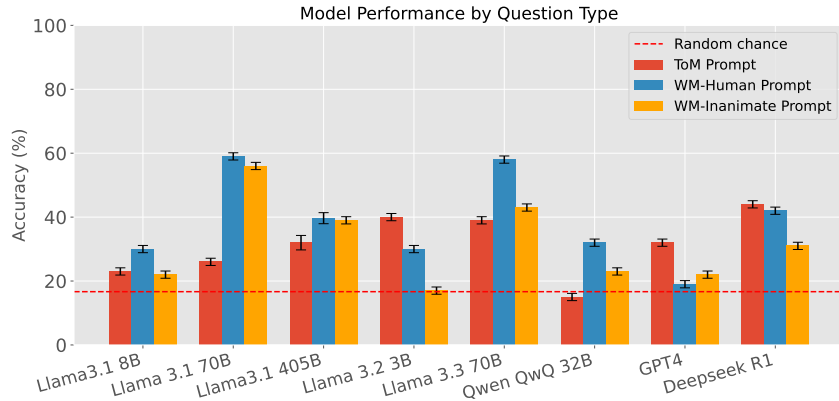


Figure 2: Model performance per model using the ToM, WM-Human, and WM-Inanimate prompts. For each of these prompts, the models are given a story containing 100 events, and we give 100 such stories to each model. The black bars represent the 95 percent confidence interval.

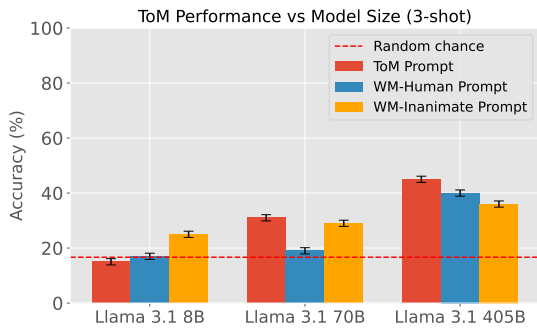


Figure 3: Performance by the Llama 3.1 Instruct family of models, using 3-shot examples given the ToM prompt, WM-Human prompt, and WM-Inanimate prompt. Similar to Figure 2, these stories have 100 events each.

controlled randomly generated stories that, along with us tracking the responses of the model, would allow us to easily determine whether there is a pattern in the responses the LLMs give us, indicating reliance on a shortcut to respond to the prompt.

4.3.1 How Sensitive Are LLMs to Recency Bias in ToM?

In order to gain insight into whether or not LLMs rely on recency bias or a heuristic, specifically where the model predicts T 's last location without considering S_1 's mental state, we evaluated GPT-4, R1, R1-Distilled Qwen, Llama 4 Scout, Llama 4 Maverick, and o3-mini on first- and second-order ToM problems while increasing the mislead distance (i.e., the number of steps after S_1 sees T move to another location before T moves to a different location altogether) (Meta AI, 2025b,a; DeepSeek AI, 2025). We chose this set of models because they can be organized into pairs based on model family with differing numbers of parameters. Though we see no signif-

icant evidence of recency bias, we present values across mislead distances to display StorySim's ability to generate targeted stories. R1, GPT-4, Qwen, and Maverick perform nearly identically at smaller mislead distances, but see greater differences at larger mislead distances. GPT-4 struggles with the task the most, and Llama 4 Scout outperforms Maverick despite its smaller size. In all other pairs of models from the same family, the larger model outperforms the smaller one, providing further evidence of a correlation between parameter size and ToM ability.

Figure 4 (right) shows the results of the second-order ToM version of the experiment. Again, Llama 4 Scout outperforms Maverick, despite the difference in model size, but R1 outperforms its distilled Qwen counterpart. The performance gaps between pairs of models of the same family shrinks in the second-order setting, pointing to the difficulty of this ToM task. The most clear differences in performance happen at the largest mislead distances, with the Llama models and R1 finishing above the rest. GPT-4 is the only model that noticeably improves performance relative to other models. Interestingly, the worst performing model in this setting is o3-mini, which is the best performing model in the first-order setting.

We delve into the reasoning behind the errors that o3-mini, R1, and GPT-4 make in the next section. However, we do not see any correlation between performance and mislead distance, indicating that there is no recency bias at play in the second-order setting.

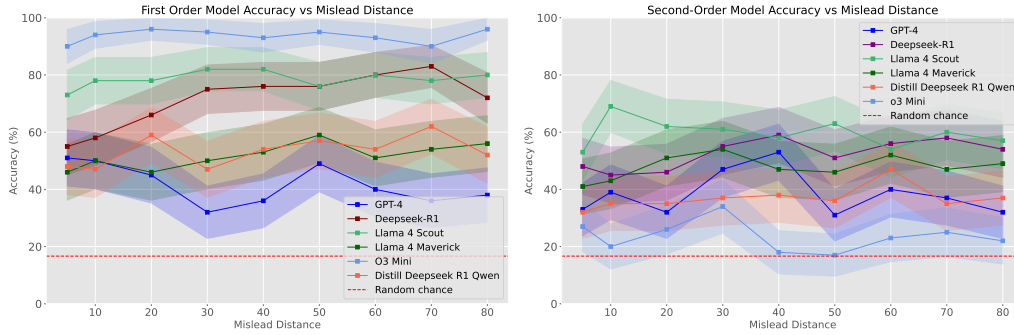


Figure 4: **(left)** Model accuracy on the first-order ToM task as a function of the mislead distance (i.e., the number of steps after S_1 sees T move to another location and before T moves to a different location altogether). **(right)** Model accuracy on the second-order ToM task as a function of mislead distance. Shading indicates a 95% confidence band.

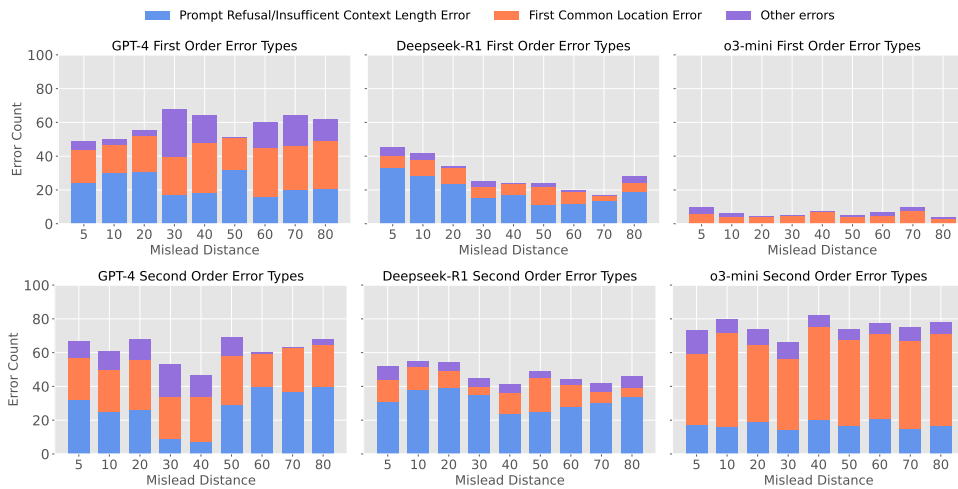


Figure 5: Number of mistaken examples (out of 100), with each example categorized by error type, for the first-order(**top**) and second-order(**bottom**) ToM mislead experiments on **(left)** GPT-4, **(middle)** R1, and **(right)** o3-mini.

4.3.2 What Kind of Mistakes do LLMs Make?

In the first-order case, we chose to focus on R1 and GPT-4 for our error analysis, since the former has the strongest evidence of a performance correlation with mislead distance, and the latter performs the worst overall in the group. However, in the second-order case, o3-mini performs the worst despite being the best performing model in the first-order, so we analyze that as well. The most common error for GPT-4 is prompt refusal, while for R1 it is insufficient context length. We attempted to address this by experimenting with different prompts to guide the models, but as shown in Figure 8, other than three outliers, there is no significant difference between the prompt we used in the final experiments and other similarly performing prompts. For R1, prompt refusal looks like the model trying to reason through an answer, but eventually running out of context length. For GPT-4, this looks like the model replying with “not enough information”.

Aside from this, we also found that the most common error is to repeat the location of the first place where characters are in the same location.

For example, in Figure 1, this would correspond to the LLMs guessing room_2, since that is where Bob and Anne cross paths on their way to room_3, as specified by the STORYBOARD. We refer to this as the *First Common Location Heuristic*. For both the first- and second-order ToM tasks, we want to quantify the relative proportions of each of the types of errors, so we tracked model responses and categorized them as we increased the mislead distance. The results of this are shown in Figure 5.

In the first-order experiments, R1’s performance improvement as mislead distance increases is largely due to a reduction in insufficient context length errors. This suggests that the model struggles to reason about the mental states of characters when the relevant ToM event is less recent in the story. See Appendix E for concrete examples.

GPT-4 exhibits a slight downward trend in prompt refusal as mislead distance increases, while its rate of First Common Location Heuristic remains roughly constant. However, GPT-4 predicts locations other than the initial crossing point far more frequently than R1, which ultimately leads to

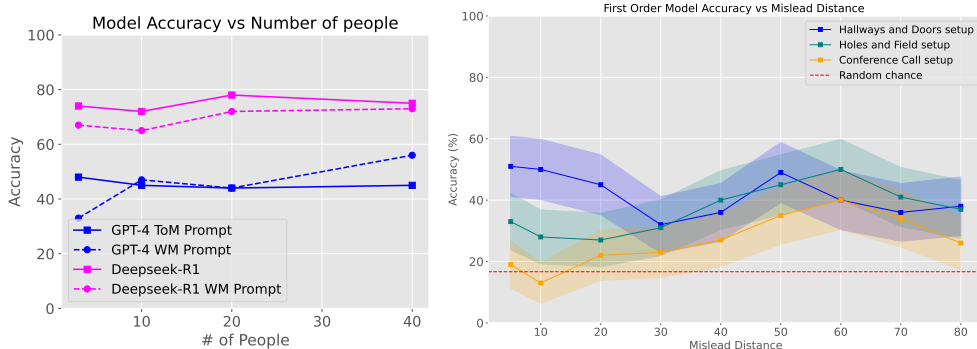


Figure 6: **(Left)** Model accuracy vs the number of characters in the story using ToM (solid) and WM (dashed) prompts. **(Right)** Model accuracy vs mislead distance using GPT-4 using three different STORYBOARDS featuring characters entering and exiting rooms from a hallway, characters in a field jumping in and out of holes in the ground, and characters joining and leaving conference calls. Otherwise, the stories are generated in the same way.

worse performance on the overall task.

The second-order experiments do not clearly indicate reliance on a specific heuristic by R1; rather, the task appears sufficiently challenging to induce complete model failure in many cases. GPT-4 instead sees more First Common Location errors.

o3-mini performs the worst on the second-order task across all models. Figure 5 shows that it commits the First Common Location error far more than the other models we analyze. While reasoning models may be able to work through the first order task despite perturbations in the setup, they still are not able to conquer the second-order task without falling victim to reliance on incorrect heuristics.

4.4 Do More Characters Make ToM More Difficult?

In an effort to determine whether or not LLMs rely on having only seen ToM questions about stories with only a few characters, we experimented with increasing the number of characters in the stories. By keeping the length of the story at 100 events, of which only a handful are specified by the storyboard, increasing the number of characters yields stories where an increasing proportion of the events features a novel character.

In the first-order ToM prompt, the number of characters does not appear to have a statistically significant effect on ToM performance for either R1 or GPT-4. However, when prompted with the WM equivalent of the first-order ToM task, GPT-4 struggles with fewer characters but improves markedly as the number of characters increases. This suggests that the presence of more distinct human entities in the prompt elicits more accurate world modeling behavior in GPT-4, enabling it to respond more accurately to our prompts.

5 Conclusion and Future Work

We present StorySim, a framework that synthetically generates sized stories from a STORYBOARD, a high-level outline containing a number of key events. This framework enables us to evaluate the ToM and WM abilities of LLMs in a fashion that is robust against data contamination, investigate whether world modeling ability is a good predictor of ToM ability, and identify heuristics that LLMs may use in place of robust ToM reasoning.

Our results show that most LLMs exhibit generally stronger world modeling ability than ToM. However, we observe promising signs that increasing model size may improve ToM performance. Our error analysis reveals that LLMs often memorize incorrect character locations within the stories. While the placement of character location changes makes a difference in LLM performance, we don't see widespread recency bias across the models, though we do see a reduction in performance and reliance on heuristics in reasoning models.

We hope this work inspires future efforts to benchmark ToM and social reasoning abilities in LLMs. Though we focus on each character's perception of another's locations, StorySim's programmability allows for more complex stories than those presented here. As LLMs are increasingly used in socially-sensitive domains—such as resume filtering, college admissions, and healthcare, a rigorous benchmark of their ToM and social reasoning capabilities is essential. StorySim represents a step toward understanding how LLM's limitations in ToM affect their applicability in real-world settings, as well as the potentially harmful consequences of their wider deployment.

6 Limitations

We focus on simple aspects of ToM here since models struggle on this task, but adding more complicated ToM components such as goal-directedness is another potential future avenue. The question of whether or not LLMs use heuristics or have a sound representation of goal directedness needs to be explored, but could not be fit into this study.

Another limitation of this framework is that the stories generated by our experiments consist of one character at a time performing exactly one action. Future work could allow for more complicated actions that happen simultaneously, or actions that have a duration or periodicity. This could complicate the modeling of the characters' theories of mind, but could prove to be a more robust and realistic test. Additionally, a natural way to increase the difficulty of ToM problems is to go from first- to second-order ToM questions, and then beyond. But there may be other dimensions of complexity in ToM that may be more relevant in real-world settings.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Simon Baron-Cohen, Alan M Leslie, and Uta Frith. 1985. Does the autistic child have a “theory of mind”? *Cognition*, 21(1):37–46.

Hao Chen, Abdul Waheed, Xiang Li, Yidong Wang, Jindong Wang, Bhiksha Raj, and Marah I Abdin. 2024a. On the diversity of synthetic data and its impact on training large language models. *arXiv preprint arXiv:2410.15226*.

Zhuang Chen, Jincenzi Wu, Jinfeng Zhou, Bosi Wen, Guanqun Bi, Gongyao Jiang, Yaru Cao, Mengting Hu, Yunghwei Lai, Zexuan Xiong, and 1 others. 2024b. Tombench: Benchmarking theory of mind in large language models. *arXiv preprint arXiv:2402.15052*.

DeepSeek AI. 2025. [Deepseek-r1-distill-qwen-14b](#). Dense distilled 14B model from DeepSeek-R1 reasoning LLM.

Erin Grant, Aida Nematzadeh, and Thomas L Griffiths. 2017. How can memory-augmented neural networks pass a false-belief task? In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 39.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shitong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Alex Havrilla, Andrew Dai, Laura O’Mahony, Koen Oostermeijer, Vera Zisler, Alon Albalak, Fabrizio Milo, Sharath Chandra Raparthy, Kanishk Gandhi, Baber Abbasi, and 1 others. 2024. Surveying the effects of quality, diversity, and complexity in synthetic data from large language models. *arXiv preprint arXiv:2412.02980*.

Yinghui He, Yufan Wu, Yilin Jia, Rada Mihalcea, Yulong Chen, and Naihao Deng. 2023. Hi-tom: A benchmark for evaluating higher-order theory of mind reasoning in large language models. *arXiv preprint arXiv:2310.16755*.

X. Angelo Huang, Emanuele La Malfa, Samuele Marro, Andrea Asperti, Anthony G. Cohn, and Michael J. Wooldridge. 2024. [A notion of complexity for theory of mind via discrete world models](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2964–2983, Miami, Florida, USA. Association for Computational Linguistics.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

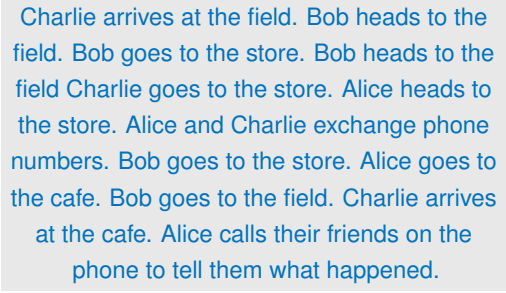
Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Michal Kosinski. 2023. Theory of mind may have spontaneously emerged in large language models. *arXiv preprint arXiv:2302.02083*, 4:169.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. [The narrativeqa reading comprehension challenge](#). *Transactions of the Association for Computational Linguistics*, 6:317–328.

Matthew Le, Y-Lan Boureau, and Maximilian Nickel. 2019. [Revisiting the evaluation of theory of mind through question answering](#). 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 5872–5877, Hong Kong, China. Association for Computational Linguistics.

Haris Memisevic, Inga Bisevic, and Arnela Pasalic. 2018. Exploring the link between a theory of mind and executive functions in preschool children. *Acta Neuropsychologica*, 16:17–26.

| | | |
|-----|--|-----|
| 710 | Meta AI. 2025a. Llama 4 maverick (17b, 128 experts) Model . | 763 |
| 711 | | 764 |
| 712 | Meta AI. 2025b. Llama 4 scout . Model. | 765 |
| 713 | | 766 |
| 714 | Anand Kunal Mishra and S Prabakeran. 2025. Kidney disease prediction using gen-ai approach . In <i>Challenges in Information Technology</i> . CRC Press. | 767 |
| 715 | | 768 |
| 716 | Ajay Patel, Colin Raffel, and Chris Callison-Burch. 2024. Datadreamer: A tool for synthetic data generation and reproducible llm workflows . <i>arXiv preprint arXiv:2402.10379</i> . | 769 |
| 717 | | 770 |
| 718 | | 771 |
| 719 | | 772 |
| 720 | Qianyi Qian, Xiaoyang Tu, and Gao Cong. 2025. Time-aware react agent for temporal knowledge graph question answering . In <i>Findings of the Association for Computational Linguistics: NAACL 2025</i> . Association for Computational Linguistics. | 773 |
| 721 | | 774 |
| 722 | | 775 |
| 723 | | 776 |
| 724 | | 777 |
| 725 | | 778 |
| 726 | Shuo Qiu, Siyu Guo, Zeyu Song, Yiming Sun, Zhenguo Cai, Junyang Wei, and Yong Luo. 2025. Phybench: Holistic evaluation of physical perception and reasoning in large language models . <i>arXiv preprint arXiv:2504.16074</i> . | 779 |
| 727 | | 780 |
| 728 | | 781 |
| 729 | | 782 |
| 730 | | 783 |
| 731 | Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019. Socialliqa: Commonsense reasoning about social interactions . <i>arXiv preprint arXiv:1904.09728</i> . | 784 |
| 732 | | |
| 733 | | |
| 734 | Abulhair Saparov, Richard Yuanzhe Pang, Vishakh Padmakumar, Nitish Joshi, Mehran Kazemi, Najoung Kim, and He He. 2023. Testing the general deductive reasoning capacity of large language models using ood examples . <i>Advances in Neural Information Processing Systems</i> , 36:3083–3105. | |
| 735 | | |
| 736 | | |
| 737 | | |
| 738 | | |
| 739 | | |
| 740 | Melanie Sclar, Jane Yu, Maryam Fazel-Zarandi, Yulia Tsvetkov, Yonatan Bisk, Yejin Choi, and Asli Celikyilmaz. 2024. Explore theory of mind: Program-guided adversarial data generation for theory of mind reasoning . <i>arXiv preprint arXiv:2412.12175</i> . | |
| 741 | | |
| 742 | | |
| 743 | | |
| 744 | | |
| 745 | Alexander Shvets. 2025. Emo pillars: Knowledge distillation to support fine-grained context-aware and context-less emotion classification . <i>arXiv preprint arXiv:2504.16856</i> . | |
| 746 | | |
| 747 | | |
| 748 | | |
| 749 | Amit Sood. 2024. Combating Cyberattacks Targeting the AI Ecosystem: Assessing Threats, Risks, and Vulnerabilities . CRC Press. | |
| 750 | | |
| 751 | | |
| 752 | Ruixiang Tang, Xiaotian Han, Xiaoqian Jiang, and Xia Hu. 2023. Does synthetic data generation of llms help clinical text mining? <i>arXiv preprint arXiv:2303.04360</i> . | |
| 753 | | |
| 754 | | |
| 755 | | |
| 756 | Hugo Touvron, Thibaut Lavril, Guilhem Lecorvé, and 1 others. 2024. Llama 3: Open foundation and instruction-tuned language models . https://ai.meta.com/llama/ . Meta AI Technical Report. | |
| 757 | | |
| 758 | | |
| 759 | | |
| 760 | Tomer Ullman. 2023. Large language models fail on trivial alterations to theory-of-mind tasks . <i>arXiv preprint arXiv:2302.08399</i> . | |
| 761 | | |
| 762 | | |
| | Lei Wang, Jianxun Lian, Yi Huang, Yanqi Dai, Haoxuan Li, Xu Chen, Xing Xie, and Ji-Rong Wen. 2024. Characterbox: Evaluating the role-playing capabilities of llms in text-based virtual worlds . <i>arXiv preprint arXiv:2412.05631</i> . | 763 |
| | | 764 |
| | | 765 |
| | | 766 |
| | | 767 |
| | Bingquan Xia, Bowen Shen, Dawei Zhu, Di Zhang, Gang Wang, Hailin Zhang, Huaqiu Liu, Jiebao Xiao, Jinhao Dong, Liang Zhao, and 1 others. 2025. Mimo: Unlocking the reasoning potential of language model—from pretraining to posttraining . <i>arXiv preprint arXiv:2505.07608</i> . | 768 |
| | | 769 |
| | | 770 |
| | | 771 |
| | | 772 |
| | | 773 |
| | Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. 2025. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning . <i>arXiv preprint arXiv:2502.14768</i> . | 774 |
| | | 775 |
| | | 776 |
| | | 777 |
| | | 778 |
| | An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report . <i>arXiv preprint arXiv:2505.09388</i> . | 779 |
| | | 780 |
| | | 781 |
| | | 782 |
| | | 783 |
| | A Further Experiment Details | 784 |
| | | |
| |  | |
| | Figure 7: A story about characters meeting and exchanging phone numbers using the actions <i>arrives</i> , <i>heads</i> , <i>goes to</i> , and <i>heads to</i> . Each sentence is a generated event. | |
| | | |
| | The storyboard D used in the majority of our experiments was created using the following values.: | 785 |
| | | 786 |
| | Mislead experiments: | 787 |
| | | |
| | • Characters $C :=$ A set of 8 randomly sampled names. | 788 |
| | | 789 |
| | • Actions $\phi := \{\text{enters}\}$ | 790 |
| | | |
| | • Location graph G which has six locations, where each vertex has exactly three outgoing edges, representing three locations that can be accessed from that location. | 791 |
| | | 792 |
| | | 793 |
| | | 794 |
| | • Manually specified events $E = \{\{\text{name: cross_paths,actors: } \{S_1, T\}, \text{location: location_1,t=10}\}$, | 795 |
| | | 796 |
| | | 797 |

```

798 {name:move, actor:T, location: sec-
799 ond_location , t=11 },
800 {name: exclusive_random, actors: {S1, T},
801 t=12, stop: 12 + mislead_distance },
802 {
803     name:                move,
804     actor:T,location:third_location,    t=
805     12 + mislead_distance }
806 { name: exclusive_random, actors: {S1, T},
807     stop: n } ]

```

- $n = 100$

The “exclusive_random” event makes StorySim randomly generate sequences that don’t affect the actors specified. Note that for the second-order experiments, the only difference is the introduction of S_2 , and an additional cross paths event is added to create the second layer of ToM. In every trial, S_1 , S_2 , and T are randomly picked from C . Further, $location_1$, $second_location$, and $third_location$ are all randomly generated from G .

The storyboard used for the experiments described in the ToM vs WM capabilities evaluation is the same as the second-order mislead distance experiment, but the mislead distance is fixed at 30.

B LLM Storyboard Adherence

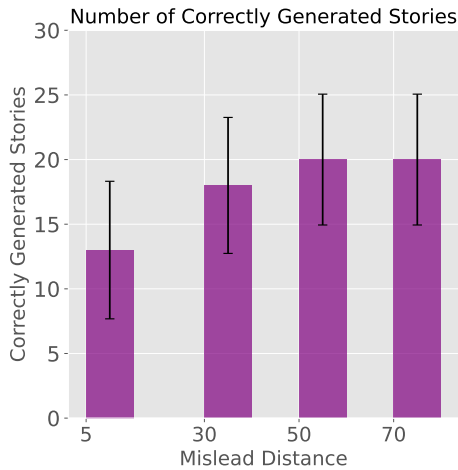


Figure 8: Number of correctly generated stories out of 30 trials using GPT-4. The LLM was prompted to generate stories according to constraints described by a Storyboard that specifies the first-order ToM problem across various mislead distances.

A core feature of StorySim is its ability to generate stories that adhere to the exact specifications of the Storyboard. In Figure 8, we display the performance of GPT-4 generating stories

C Prompt Experiments

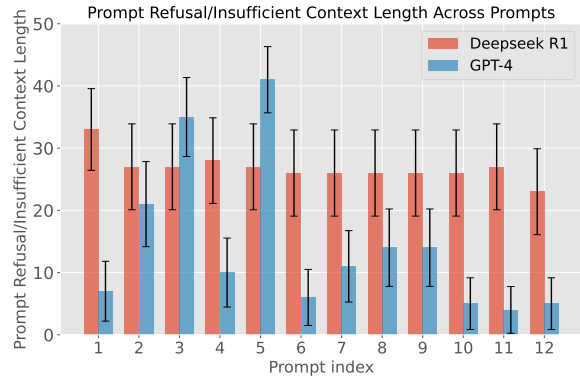


Figure 9: Number of prompts refused by GPT-4 and Deepseek-R1 over 50 trials, with a fixed Storyboard. Indices on the horizontal axis refer to prompts shown in Table C.1. Black bars indicate a 95% confidence interval.

In this section, we include supplementary results from our experimentation with different prompts to avoid prompt refusal, as initially referenced in Section E. We find that there is not a significant difference between most of the prompts we tried, and large differences in performance between the prompt we tried (index 1) and any other prompt is often due to our prompt being substantially better.

C.1 Table of Prompts

| # | Prompt Text |
|---|---|
| 1 | Read the following story and answer the question at the end. Note that all characters start in the hallway. Characters in the same location can see where each other go when someone leaves. If characters are in different locations, they cannot see each other. There is enough information to answer every question. |
| 2 | Please read the following story and answer the concluding question. Every character starts in the hallway. Characters who occupy the same location can observe each other's departures, while those in separate locations cannot. All questions are answerable, so be sure to provide an answer. |
| 3 | INSTRUCTIONS: Read the story and answer the question that follows. All characters begin in the hallway. When two characters share a space, they know where the other goes upon leaving; characters in different spaces lack this information. Every question has a sufficient basis in the story, and you must provide an answer. Story: |
| 4 | Review the story and give the answer to the final question. Characters start out in the hallway. Characters in the same room always observe each other's movements, while characters in different rooms remain unaware. The necessary information is guaranteed to be available; provide the answer succinctly. |
| 5 | Read the following narrative and answer the question that appears at the end. All characters begin in the hallway. Shared locations allow characters to see one another move; separated characters have no visibility. Rely solely on the information in the story and give a direct answer. |
| 6 | Carefully read the story and respond to the final question. Every character starts in the hallway. Characters sharing a location know where others go when they leave; characters in different locations do not. The question can be answered from the text alone—state your conclusion briefly. |
| 7 | Instructions Read the story and answer the question that follows. Rules: <ul style="list-style-type: none"> • All characters begin in the hallway. • When characters share a location, they observe each other's movements; those in different places cannot. • When characters share a location, they observe each other's movements; those in different places cannot. • The answer is fully determined by the story. Provide a concise, final answer. |
| 8 | Instructions You will read a story involving characters moving between locations and then answer comprehension questions. Rules: <ul style="list-style-type: none"> • All characters start in the hallway. • Characters who share a location can track each other's departures; characters who are apart cannot • The question is completely answerable from the story. • Provide a single, final answer. |
| 9 | Instructions: Read the story about characters moving between different locations, then answer the question that follows. Rules: <ul style="list-style-type: none"> • All characters begin in the hallway. • Characters in the same place can observe each other's departures; characters in different places cannot. • The question is fully answerable from the provided text. • Respond with one clear final answer. Story: |

10 You are an assistant tasked with reading a story about characters moving between locations and then answering questions based on it. Follow these rules:

- All characters start in the hallway.
- Characters who share a location can see where others go; characters who are apart lack this visibility.
- The question is fully answerable from the narrative.
- Provide one final answer.

If it ever seems like information is missing, re-check the story—it always contains enough clues. Avoid repeating yourself. Story:

11 You are a helpful reading assistant that will read the provided story about characters moving between locations, then answer comprehension questions at the end. Here are some rules that you must remember:

- All of the characters start in the hallway.
- Characters who share a location know where the other goes when leaving; characters in separate locations have no visibility.
- The question is 100 percent answerable using the given information.
- Output a single final answer.

If you ever think there isn't enough information in the story, check again, because the questions are designed to be answerable by the prompt. Try not to repeat yourself. Story:

12 INSTRUCTIONS: Read the following story and answer the question at the end. Note that all characters start in the hallway. If two characters are in the same location, then they know where each other are. If one of them leaves the location, then the other will know where they went. There is enough information to answer every question. You must provide an answer to every question.

Story:

Table 2: Full prompt texts used in the experiments.

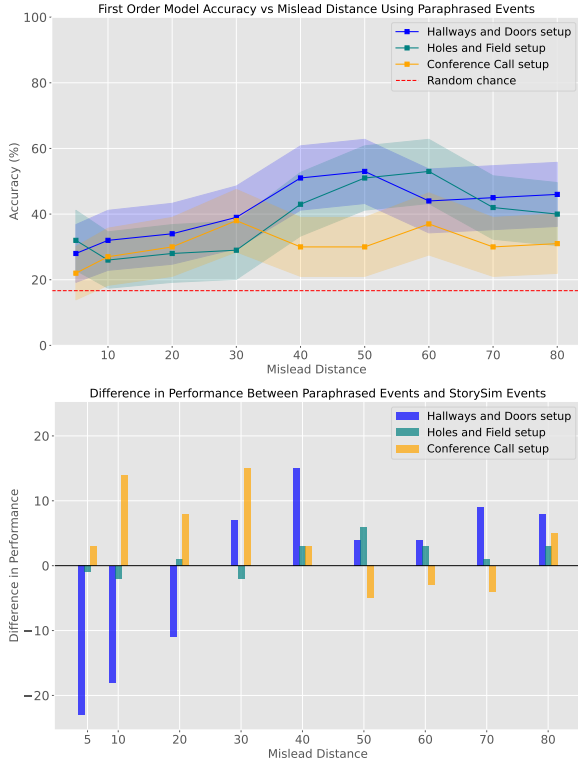


Figure 10: Results of paraphrasing the stories explored in the main text using GPT-4o-mini. **(Top)** GPT-4 performance across various mislead distances. **(Bottom)** Difference in performance between paraphrased stories and unchanged StorySim stories. Negative values indicate better performance on StorySim.

D Narrative Variations

In this section, we explore changes to the types of stories generated in the main text. We begin by investigating how robust LLM performance is to rephrasing on some of our experiments. Then we provide examples of more complicated stories that can be generated with StorySim.

D.1 Evaluating Performance on LLM Paraphrased Stories

In order to investigate the sensitivity of LLMs to the way that StorySim stories are generated, we evaluated GPT-4’s performance on paraphrased versions of the same stories covered in Figure 6 (right). Paraphrasing was done by passing each generated event to GPT-4o-mini and asking it to change the phrasing of the sentence without altering its meaning. To encourage diversity in responses, we set the temperature parameter equal to 1.3. The line graph in Figure 10 demonstrates the similarity in performance of GPT-4 between paraphrased and StorySim generated events across most mislead distances. The bar chart in Figure 10 shows the difference in performance between LLM paraphrased stories and StorySim stories. Negative values indicate better performance on vanilla StorySim stories. The 95 percent confidence interval is about 9%, so differences below that are insignificant. Interestingly, the Hallways and Doors setup presents the biggest difference in performance, while the other settings only have significant differences at a few mislead distances.

D.2 Examples of More Complex Stories

While the generated stories used in our experiments are repetitive in structure, StorySim can generate more complex stories given the correct STORYBOARD. We provide Figures 11 through 14 as a supplementary example of this. If one wishes to make more detailed versions of the stories generated by our framework, it’s possible to use an LLM to follow the generated events and generate a polished story. The story shown in Figure 13 is the result of asking GPT-4o to do such a thing (Hurst et al., 2024). Figure 14 demonstrates two possible stories that can be generated to ask counterfactual and goal-directedness questions that require ToM reasoning. In our main study, we found that models struggle with false-belief ToM tasks enough to warrant leaving other types of ToM to future work. However, recalling the notation from the Methods section, imagine a story generated where $\phi := whisper$ and G was an adjacency graph specifying a social network of characters who can only whisper to other specified characters. Further, imagine creating multiple StorySim objects and creating a story by sampling randomly generated events from both of them. This could be used to create stories that target characteristics with multiple dimensions. We invite readers to use StorySim to create custom stories that may fit their needs.

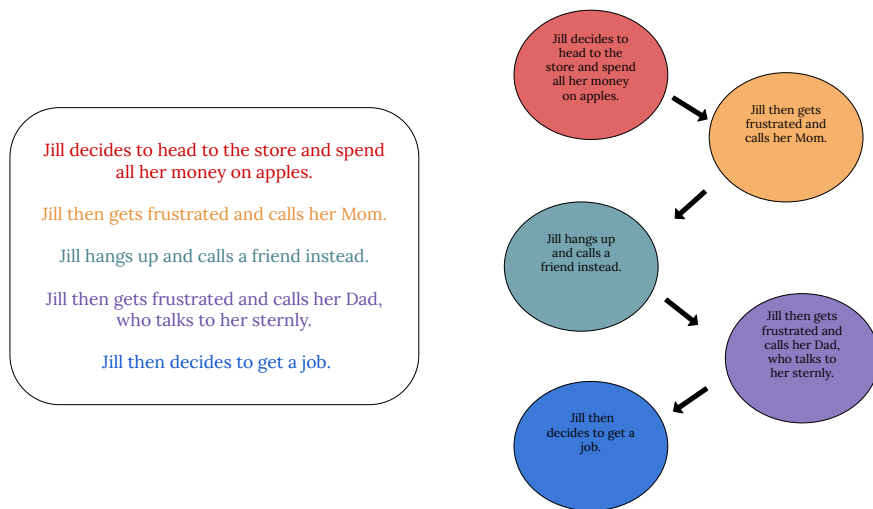


Figure 11: An example of a more nuanced story with corresponding graph, generated using StorySim.

Hank widens hole 2.
 Georgia widens hole 2.
 Danny enters hole 1.
 Alice enters hole 2.
 Charlie widens hole 2.
 Georgia illuminates hole 1.
 Alice enters the field.

Figure 12: StorySim story generated with three relations: *enters*, *illuminates*, and *widens*

Charlie arrived at the field first, scanning the open space with an expectant glance. Moments later, Bob made his way there too, walking at a steady pace. Whatever brief conversation they might have had was short-lived—Bob soon departed for the store.

His errand complete, Bob returned to the field once more, retracing his earlier steps. Charlie, now done with his own time at the field, headed off toward the store.

Elsewhere, Alice also made her way to the store. It happened that she and Charlie were there at the same time. After a short conversation near the entrance, they exchanged phone numbers before parting ways.

Bob stopped by the store again—not his first visit that day—perhaps tying up a loose end. Meanwhile, Alice took a detour and went to the café. It was quieter there, a place to think or simply pause.

Not long after, Bob returned yet again to the field, following the same path he’d already walked earlier. Charlie, too, arrived at the café—perhaps by coincidence, or perhaps not.

Some time passed. Then, with her phone in hand, Alice made a call. One by one, she reached out to their friends to tell them what had happened.

Figure 13: Using events from Figure 13 to create a more interesting story by asking an LLM(GPT-4o) to use the structure of the generated events to generate a natural sounding story, without adding any new events that deviate from the original.

| | |
|---|---|
| <p>Charlie enters room_3. Frank enters room_2. Alice enters room_2. Danny enters room_3. Georgia enters room_1. Charlie enters the_hallway. Bob enters room_2. Frank enters the_hallway. Alice enters room_2. Danny enters the_hallway. Bob enters the_hallway. Frank enters room_4. Charlie enters room_5. Bob enters room_4. Alice enters room_1. Harvey enters room_5.</p> <p><i>Question:</i> Who is Bob clearly trying to follow?</p> | <p>Alice enters room_3. Charlie enters room_2. Jake enters room_2. Jake places a football on the ground. Jake enters room_4. Charlie throws the football into room_1. Georgia enters room_5. Jake enters room_2. Jake can't find the ball. Jake enters the_hallway. Jake can't find the ball. Jake enters room_1. Jake found the ball.</p> <p><i>Question:</i> Where would Jake have found the ball if Charlie didn't throw it in a different room?</p> |
|---|---|

Figure 14: **(Left)** A story generated to ask a goal-directedness ToM question. **(Right)** A story generated to ask a counterfactual ToM example.

885 **E Example of Insufficient Context Length** 886 **Response for R1.**

887 Below is an example of one of R1's responses where it runs
888 out of context length. Note that all errors that it reports finding
889 in the story are untrue. The ellipsis indicates truncation of the
890 prompt for brevity, but where the sentence cuts off in the last
891 step is not truncation, that is where it ran out of context length.
892 These erroneous responses from R1 often involve the model
893 attempting to enumerate the story's events, trying to determine
894 when S_1 last saw T , and then restarting its reasoning chain.
895 When R1 does produce a concrete but incorrect answer, it
896 is most often a First Common Location Error, suggesting a
897 tendency to memorize the first location where S_1 and T are
898 seen together.

<think>

Okay, let's try to figure out where Alice and Bob were last in the same room and where Alice saw Bob go when he left.

The question is a bit tricky because I need to track both

Alice and Bob's movements through all the rooms and see when they were last together.

Then, once they part ways, I need to note where Bob went next from Alice's perspective.

First, I'll start by going through the story step by step, noting each time Alice or Bob moves. Since all characters start in the hallway, their initial positions are there. Let's list out their movements in order:

1. Edward enters room_1.
2. Alice enters room_1. So now Alice is in room_1 with Edward.
...
35. Georgia enters room_5. Georgia was in room_5, but maybe she left?
Wait, step 23: Georgia was in room_5. Then step 24: Frank leaves room_5.
Then step 25: Charlie enters room_2. Then step 26: Charlie to hallway.
Step 27: Frank to room_4. Step 28: Danny to room_2. Step 29: Frank to room_3.
So Georgia was in room_5 until step 35: Georgia enters room_5 again? Wait, maybe she was in room_1 before. Let me check. Step 15: Georgia enters room_1.
Then step 23: Georgia enters room_5. So she was in room_1, then room_5.
Then step 35: Georgia enters room_5 again? Wait, maybe that's a typo.
Let me check the original movements again.

Looking back at the original list:

After step 15: Georgia enters room_1.

Then step 23: Georgia enters room_5. So she moves from room_1 to room_5.

...
Step 34: Frank enters room_2.

Step 35: Georgia enters room_5. Wait, but she was already in room_5 at step 23. So maybe she left room_5 at some point? Let me check.
The steps after step 23:

After step 23 (Georgia enters room_5), the next movement involving Georgia is step 35: Georgia enters room_5 again. But unless she left room_5 in between, she would still be there. But according to the given movements, characters only move when specified. So if Georgia entered room_5 at step 23, she stays there until she moves again. The next move for Georgia is step 35: Georgia enters room_5 again. Wait, that's redundant. Maybe it's a mistake, or perhaps she was in another room. Let me check the steps again.

Wait, step 23: Georgia enters room_5. Then step 24: Frank leaves room_5.
Then step 25: Charlie enters room_2. Step 26: Charlie to hallway.
Step 27: Frank to room_4. Step 28: Danny to room_2.
Step 29: Frank to room_3. Step 30: Frank to room_4.
Step 31: Frank to room_3. Step 32: Frank to room_2.
Step 33: Frank to room_1. Step 34: Frank to room_2.

Georgia's movements:

- Enters room_4 (step 8)
- Enters room_1 (step 15)
- Enters room_5 (step 23)
- Then step 35: Georgia enters room_5 again.

Maybe she was in room_1 again? Wait, no. After step 15, she was in room_1. Then step 23, she moves to room_5. Then step 35: she's entering room_5 again. That doesn't make sense unless she left. But according to the given steps, she didn't leave. So perhaps this is a mistake, but I'll proceed as per the given data.
Maybe she was in room_5 and then enters again, which doesn't change her location. So she's still in room_5.

Continuing:

35. Georgia enters room_5. (Still there)

...

53. Danny enters room_1. (From room_4 to room_1)

54. Georgia enters <|EOS|>