



Universiteit
Leiden

Changing the Narrative: Improving NPC Dialogue Consistency Us- ing a Single LLM

Tim Verberne (s4080084)

Supervisors:

Dr. M. Preuss & Dr. M. Müller-Brockhausen

BACHELOR THESIS – DATA SCIENCE AND ARTIFICIAL INTELLIGENCE

Leiden Institute of Advanced Computer Science (LIACS)

liacs.leidenuniv.nl

July 1, 2026

Abstract

Dialogue in digital games was always static, pre-programmed, but with the rise of LLMs, new possibilities have emerged to generate NPC dialogue in real time. This study explores how to improve dialogue consistency in LLM-generated dialogue, using a single LLM for NPC dialogue. For the dialogue agent, the gpt-oss:120b model was used via Ollama, no extra models were used for state tracking. Improving dialogue consistency was approached by having three different story settings with a differing number of event states and objective states. Various ways of prompting were also used to test the influence of different types of prompting on dialogue consistency. It was found that the number of states had no significant influence on the number of consistency errors, but a higher number of event states did appear to result in a higher chance of out-of-turn errors. A validation study was done to confirm this but did not provide the data to confirm it. Also, the amount of remembered contextual and world knowledge varied depending on which prompts were used.

Contents

1	Introduction	1
2	Background and Related Work	2
2.1	Dialogue in Games	2
2.2	LLM Generated Dialogue	3
2.3	Dialogue Consistency	4
3	Method	6
3.1	Setting up the Experiment	6
3.2	The Right LLM	6
3.3	The Variables and the Stories	7
3.4	Evaluating the Results	8
3.5	The NPC Dialogue Agent	9
4	Results	10
4.1	The Story Settings	11
4.1.1	Knowledge Retention	11
4.1.2	Consistency Errors	13
4.1.3	General Metrics	16
4.2	Few Events - Few Objectives	17
4.2.1	Knowledge Retention	17
4.2.2	Consistency Errors	19
4.3	Few Events - Many Objectives	22
4.3.1	Knowledge Retention	22
4.3.2	Consistency Errors	24
4.4	Many Events - Few Objectives	27
4.4.1	Knowledge Retention	27
4.4.2	Consistency Errors	29
5	Validation Study	32
6	Discussion	34
7	Limitations	35
8	Future Research	35
9	Conclusions	35
	References	38
A	Prompt Templates	38
A.1	General prompts	38
A.2	Story 1: Few Events Few Objectives.	40
A.3	Story 2: Few Events Many Objectives	42

A.4 Story 3: Many Events Few Objectives	46
A.5 Evaluation Prompts	49
A.6 Validation Prompts	51
B Example of Complete Dialogue	54
C Examples of Each Error Type	60

1 Introduction

In modern digital action role-playing games (RPGs), most of the Non-Playable Character (NPC) dialogue is pre-programmed into the game. This means that for everyone who plays the game and every time the game is played, the conversation between the player and the NPC is the same, both in content and outcome. Sometimes developers will play with this a little bit by adding multiple playable characters or by letting players choose a dialogue option. However, this will often not change the outcome of the game and only change the NPC’s response a little. This means that not enough variability is present to create an incentive for the player to play the game again. Elements of chance can be introduced to create variation, but applying this to a narrative limits a developer’s control on the story [26]. If a developer wants real variability and remain in control, all dialogue options and story narratives have to be pre-programmed. However, this is almost never done as it significantly increases the game development time and the size of the game itself. However, with the arrival of Large Language Models (LLMs), a new possibility of generating dialogue has become available. With LLMs, the possibility is there to generate NPC dialogue in real time while playing the game, without the need for large pre-programmed dialogue trees.

At the moment, these LLMs are not consistent enough to generate the dialogue for a narrative-driven conversation based in a larger game world. They tend to hallucinate information, forget previously introduced information, or forget information about the world itself. Various methods have been introduced to improve dialogue consistency, from prompting methods to state-tracking modules, all with varying amounts of success [4]. For LLM-generated NPC dialogue in games, the goal would be to achieve a varied conversation while adhering to the narrative structure and maintaining a consistent dialogue. Our study is based on a dual-LLM framework that frames interactions as a series of milestone-based goals to ensure narrative coherence [20]. These milestones provide a game developer some control over the narrative while still allowing the LLM some room for variation.

This study will implement an NPC dialogue agent that will have conversations with a player. A single LLM will be used for the agent, without any additional LLMs or models for state-tracking. This is done to reduce complexity and latency. The conversations will happen in three different story settings: one setting with few event states and few objective states (FeFo), one with few event states and many objective states (FeMo), and one with many event states and few objective states (MeFo). Six different types of variable packages will be used to determine the influence of different types of system prompting on dialogue consistency.

The expectation is that a story with more event states gives both the developer more control over the narrative and the LLM more information to work with. While still allowing the LLM to fill in information that has not been given to it, and so enabling an environment where there is variation. More event states should thus mean a more consistent dialogue. By exploring this, the study will make the following contributions:

- **Determining the possibility of using a single LLM to generate consistent NPC dialogue.** By using only a single LLM, the load and complexity for future implementations are kept at a minimum.
- **Determining the influence of the number of event states and objective states on dialogue consistency.** By analyzing three different story settings, each with a varying

number of event states and objective states, we aim to determine the influence of the amount of such states on the dialogue consistency.

- **Determining the influence of different types of system prompts on the dialogue consistency.** By using different types of system prompts, we aim to determine the influence on dialogue consistency.

In this paper, we start by looking at the background of this topic and discussing some of the related work in Chapter 2. After this, we will explain the methodology of the experimental study in Chapter 3. The results of the experiment will be provided in Chapter 4. A validation study was performed after the main study, the setup and results of this study are discussed in Chapter 5. These results will be discussed in Chapter 6. Reflecting on the study will be done in Chapter 7 by looking at the limitations. In Chapter 8 we will set out some options for future research to improve upon what was done in this study. At the end, we will provide our conclusions in Chapter 9.

2 Background and Related Work

This section will first provide a short background on dialogue in games in 2.1. Then we will provide a good overview of current work related to improving dialogue consistency. We will look at the possibility of generating dialogue using LLMs in 2.2. Then we will look at various possibilities of how the quality of this dialogue can be improved in 2.3.

2.1 Dialogue in Games

Games, by their very nature, are something that requires an interaction of some kind. In their most simplistic form this could just be by means of flipping a card, adding a line, or writing down a number, for example when playing a game of Solitaire or a Sudoku. When playing multiplayer games this interaction expands, both with the game itself and with other players. Discussions about tactics, strategies, trades and so on, enrich the gameplay and with this dialogue, elevate the gaming experience itself. Some games add to this by having action cards or challenges that initiate events or prompt the player to act. Having this can greatly increase the replayability of a game, by having each play-through be unique.

One of the earliest, most popular and still widely played games that is a great example of this type of gameplay is *Dungeons & Dragons* (D&D). A fantasy tabletop role-playing game (TTRPG). In the game, played with a group of people, a 'Dungeon Master' sets out quests for the other players to play through. The Dungeon Master guides the players through the game by telling and setting the story. The Dungeon Master will also impersonate any friend or foe, when encountered, that inhabits the game world, the NPCs. This way, a game can be tailored exactly to the players, and is every game a unique experience. Some of these games, or 'campaigns', can last several years, while the story progresses, the game and its narrative evolve.

When games started to become digital, some of this flexibility was lost. Digital action cards or challenges were still an option, however impersonating NPCs like in a TTRPG was no longer possible. Now, every action, every word spoken by an NPC in a game, had to be pre-programmed. If developers wanted dialogue options, this would mean huge dialogue trees where every choice would be a new branch. This explosion in size meant that most story-driven games ended up

having a single storyline, where every play-through had the same dialogue utterances and the same outcome. Sometimes players were given the illusion of choice, having the possibility to choose between two or three dialogue options. However, this would often have little to no influence on the further course of the conversation or the story.

Now, with the rise of Large Language Models (LLMs), a new possibility for dialogue has arrived. By generating dialogue while playing, a similar player-tailored experience to those of the TTRPS can be achieved. Without having to pre-program the entire dialogue of a conversation.

2.2 LLM Generated Dialogue

In recent years the field of LLM-generated stories, conversations and dialogue has grown much. Exploring the possibilities and limitations of LLMs. In this section, we look at some frameworks that were created for LLM-generated stories and case studies on people playing games with LLM-driven NPCs.

Word2World is a framework that uses the zero-shot capabilities of LLMs to extract character and environmental details from text. This was done to generate a playable 2D tile-based game. In the study a multi-step pipeline was used to design a story setting which is improved using a feedback loop to ensure consistency [13]. Another story agent framework is *Words to Worlds*, this framework is a pipeline that can transform a prompt into story. It achieves this by letting an LLM break down a narrative into hierarchical components. By doing this the framework aims to improve narrative consistency [24]. Another framework is *SceneCraft*, this is a framework that generates narrative scenes. It automates NPC interactions important to key moments in a story. It can generate individual game scenes aligned with the intentions of the developers. It does this by analyzing scene objectives, NPC traits, location, and narrative. The framework uses semantic extraction to automatically assign appropriate facial expressions and physical gestures to NPCs. This allows it to generate dialogue that adapts to a player’s choices while adhering to interaction goals [8]. KNowledge Constrained User-NPC Dialogue GEneration (KNUDGE) is a dataset of NPC dialogue trees. It is designed to improve the automatic generation of dialogue trees for NPCs. Using an LLM a method was developed that uses in-context learning to generate new dialogue that is game-specific and true to the game lore [29].

LIGS or the LLM-Infused Game System is a prototype game used to observe gameplay experience and progression. It uses an LLM to determine NPC stories and update world data. Here, an important finding was that a lack of guidance and structure can negatively impact a player’s gaming experience. It was also found that while freedom enhances narrative space it can also challenge decision-making, and generated content allows for flexibility but can result in a broken narrative [7]. Another case study into LLM-powered NPCs highlighted the importance of NPCs to a game world and that LLM-powered NPCs add more depth by allowing for more natural conversations. A challenge on the other hand was that, by their nature, LLM-powered NPCs are more likely to go off-script and break the immersion. However, the majority of participants still found the LLM-powered NPCs a positive influence on the game’s re-playability [27]. Another study compared different story generation methods against human-written stories. For this, GPT-3 and prompt-based learning were used to generate high-quality stories. It found that the quality of LLM-generated stories were significantly higher compared to those made by other generation methods. These stories were also often on a comparable level to human-written stories in terms of fluency and coherence. However, there were still issues, as there were still many generation errors

and the LLM tended to 'soft plagiarize', as it had the tendency to copy directly from source material [30]. Another study examined the impact of LLM-generated narratives in a visual novel, specifically a serious game for educational purposes. Here, one story used thematic keywords in the prompts and the other did not. It was found that the narrative engagement was higher for the group without the keywords, but the experience satisfaction was similar [6]. Another study looked at the possibility of addressing the narrative paradox with LLM-driven NPCs. The narrative paradox is the struggle between finding the balance of maintaining a structured, coherent story and allowing the user free choice. Here, players of a game with LLM-driven NPCs had an overall positive experience of the game. Participants perceived themselves to be at least somewhat to fully in control, and the narratives were also judged to be somewhat to fully coherent [19]. Based on LLM-generated content for NPCs, a case study was done to provide guidelines for developing LLM-driven NPCs. Player feedback was gathered from online platforms and analyzed. The guidelines that were then developed touch on things like hallucinations, conversational content, conversational memory, conversational style, conversational direction, NPC evasiveness, and NPC social cues [4].

Another novel idea is to have NPCs not just act with players inside the game, but also on other platforms outside of it. A study introduced a cross-platform NPC powered by an LLM. Here the NPC can engage with the player both inside the game and on social platforms. For this, dialogue logs were stored in a cloud database to allow for cross-platform interactions. The responses were adapted based on the context and the platform from which the message originated. This was all text-based and long-term memory retention is still an issue [25].

2.3 Dialogue Consistency

While there is already a lot possible, the largest issue is still dialogue consistency. In this section, we explore a few options to improve dialogue consistency: various frameworks, prompting techniques, and state tracking.

Shot prompting is when an example of how to reason or to respond is given in the prompt. Zero-shot prompting is when this is not done, one-shot prompting is when a single example is given and few-shot prompting is when more than one example is given. One of the most famous prompting techniques is Chain-of-Thought(CoT) prompting. Here, a few examples are provided as chain-of-thought demonstrations. These one-shot or few-shot prompting techniques have been shown to significantly improve reasoning capabilities in LLMs [28]. Another proposed prompting technique is *Self-Explanation Prompting*, a zero-shot prompting method. The idea of this technique is to let the LLM analyze each dialogue utterance before execution, this gives the LLM a moment to reflect and rethink, and thereby improving performance [5]. Another prompting method uses *Deflanderization* to suppress excessive role-play and improve dialogue quality. The term "deflanderization" is used to explain something in plain language. So when using this method, you ask the model to respond naturally and concisely while avoiding exaggerated role-playing [3]. Another proposed prompting technique is to generate prompts based on instance-level control codes instead of conversational history. This controlled dialogue prompting provides attribute information within the prompt to steer the model [12].

Another method of improving dialogue consistency is by using additional models or methods for state tracking, or the usage of state graphs. So can dialogue state tracking be used to dynamically track information within dialogue states. A Hierarchical Fine-Grained State Aware Graph Attention Network (HFSG-DST) was used to do this. It models both dependency and constituent trees to

capture fine-grained linguistic structures. It also makes a schema-enhanced dialogue history selector. Using this, a higher performance is achieved by reducing noise and emphasizing key grammatical entities [10]. Another study introduced a dual-perspective state representation approach. This approach provides a dynamic and inductive method for topological state representation learning in episodic reinforcement learning within task-oriented dialogue systems. The model extracts inherent topological information from state transitions in the Markov Decision Process graph by employing a modified clustering technique to address the limitations of transductive graph representation learning. It inductively captures structural relationships and enables generalization to unseen states. It also incorporates dynamic graph representation learning with task-specific rewards using temporal difference error. This captures topological features of state transitions, allowing the system to adapt to evolving goals and enhance decision making [23].

A framework that partly inspired this research is on goal-oriented interactions. In this framework, two separate LLMs are used, one to provide immediate responses, the dialogue LLM. The other is the question-answering LLM, this LLM assess the situation and judges if the predetermined criteria for state transitions have been met. This state tracking system allows for a more consistent dialogue in a goal-oriented conversation [20]. *SNAP* or Story and Narrative-based Agent with Planning is an LLM-based conversational agent. This framework structures narrative context into cells with plans to prevent a story from deviating from the narrative. The cells provide scope limitation, progress tracking, and intervention points. Plans are made by a separate planning model that ensures quality via a filter, and a separate agent model then generates the dialogue [1]. Another framework is *D-SMART*, which is designed to maintain multi-turn dialogue consistency by enabling LLMs to build and reason over a dynamic, structured representation of the conversational context. This is achieved by a dynamic, structured memory component that constructs and maintains a knowledge graph of the conversation. And a reasoning tree that executes inferences as a search over the graph [9]. Another method to improve conversation quality in LLM-driven NPCs is by adding contextual awareness of their surroundings. This was done by adding panoramic images of the NPC’s surroundings and applying semantic segmentation to identify objects and their positions. This resulted in a more immersive gameplay where the NPC knows more about its environment and feels like it’s more part of the world [22].

Another module, made specifically for the werewolf game, an incomplete knowledge game, uses 4 separate modules to manage dialogue consistency. A situation analysis module uses an LLM to take the rules and dialogue history and process these for the current game situation. A response generation module, an LLM that takes the output of the situation analysis module and, together with specific prompts, provides a dialogue utterance. A persuasive response module tries to assess other NPCs’ dialogue and give a response in a specific category. A voting module then handles the voting specific to the game. With these models the dialogue is managed to stay consistent [21].

A large challenge of incorporating LLMs into games is their size, hardware requirements, and possible latency problems. A modular memory model was proposed to use Small Language Models (SLMs) for NPC dialogue. The memory modules can be swapped at runtime and preserve character-specific conversational context and world knowledge. Results have shown that, while not faultless, it is promising [2].

Another proposed method to keep the dialogue consistent is to use narrative patterns to maintain thematic consistency in the composition of stories. By comprising the pattern-guided composition of narratives and the most specific generalization criterion for new patterns LLMs were able to create complex narratives [11].

3 Method

In this chapter, the setup and method of the research are explained. In section 3.1, the general structure of the experiment is laid out. In section 3.2, the selection process and functionality of the LLM that was used are explained. In section 3.3, we explain how we developed the variable packages and the three story settings used for prompting. In section 3.4, it is explained how the results will be evaluated. Finally, in section 3.5, the structure of the NPC dialogue agent is laid out.

3.1 Setting up the Experiment

The main idea of the experiment is as follows. In order to assess how to improve dialogue consistency, a dialogue agent is made. This NPC dialogue agent is an LLM that can have a multi-turn dialogue with a player, using text messages. A manual way of including or excluding certain prompt packages from this agent is used to study the effects of various methods of prompting on dialogue consistency and narrative. In order to study the effects of the number of event states and objective states on the dialogue consistency, three story settings were created. One with few event states and few objective states (*FeFo*), one with few event states and many objective states (*FeMo*), and one with many event states and few objective states (*MeFo*). The prompts that were used to construct these story settings can be found in Appendices A.2, A.3 & A.4.

To evaluate dialogue consistency and narrative, two things will be measured. First, during the conversation, after every 5 dialogue turns, an evaluation prompt about context retention and world knowledge retention is asked outside the main dialogue loop. The prompts that were used for this evaluation can be found in Appendix A.5. The number of correctly remembered context statements and world knowledge will be manually counted from the LLM’s response to the evaluation prompts after the conversation is over. Secondly, consistency errors will also be counted manually after the conversation is done. However, these are counted at every dialogue turn instead of after every five turns.

3.2 The Right LLM

For the NPC dialogue agent, an LLM had to be chosen. The preferred first choice was to use OpenAI’s latest version of ChatGPT with an API in order to have the largest amount of flexibility and control [18]. However, since this is behind a paywall, other options were explored. Other state-of-the-art models were considered, but these also had their API functionality behind a paywall. This led us to explore the option of open models, which limited the LLM options. Open-source LLM models can be downloaded for free to run on private devices or can be run on external servers via cloud or API services. We went with Ollama, a provider that allows people to download open models to run locally, but also has options to run certain models on external servers via API or via cloud [14]. We chose to run the NPC dialogue agent via API calls, so we would not be limited by our own hardware. There was also the option to run the model via Ollama’s cloud services, this was not chosen as this option did not appear to have any significant edge over the API option.

There was still a preference to use a model from OpenAI’s GPT family, because these models are some of the most studied and best-performing models. However, the latest versions of ChatGPT or other state-of-the-art models are not open models. However, some models of the GPT family are open models and available on Ollama. OpenAI’s four gpt-oss models were evaluated, and after some

consideration, gpt-oss:120b was chosen. This model was chosen because it should perform better than its smaller 20b variant, just by its size alone. It also has similar performance to OpenAI’s o4-mini [17, 15]. The gpt-oss-safeguard models were also considered, however this study was found not to be the right use case for these models [16].

3.3 The Variables and the Stories

Three different story settings were made for different event state and objective state settings. The *FeFo* story setting is an interrogation setting, where the player is a detective, and the NPC is impersonating someone who is being questioned. This setting was chosen because the events are only two event states, or things the detective needs to determine, and the objective states are either guilty or not guilty. The *FeMo* story setting is a world where the player is the president of a fallen nation. With the help and advice of the head of security, the NPC, the player needs to flee and get to a safe location. This setting was chosen because there is little information given on how to flee the country, but there are a lot of options to flee. Resulting in only three event states and seven objective states. The third story setting, *MeFo*, is a setting of two astronauts working on repairs on the outside of a spaceship. This setting was chosen as there are a lot of things to fix, but the only outcomes are fixing the ship or not. Giving a total of seven event states and two objective states. The complete overview of these story settings, including evaluation prompts and instruction prompts, can be found in Appendix A.

The prompting variables are based on previous work and earlier established guidelines, sections 2.2 & 2.3. Chain-of-Thought (CoT) prompting will be used, and all prompt parts will be based on the guidelines for developing LLM-Driven NPCs [4]. The variables are ordered in six different variable packages; *BASE*, *KNOW*, *STATE*, *REASON*, *CHAR* and *ALL*. Here *BASE* is always included in the prompt, the other packages can be added to this. The *ALL* variable package is the package where all packages are included. If *KNOW* is used, what is meant is that the two unique variables from the *KNOW* package are added to the *BASE* package. The basic variables are always included in the *BASE* package; *base persona* for the basic character instructions for the NPC dialogue agent, *world setting* for the basic story setting, *event states*, which list all the event states for the story setting, *objective states*, which list all the objective states for the story setting and *style instructions*, which provides the basic guidelines for the answer. None of the variables in this package are story-specific. The *KNOW*, or knowledge pack, provides the LLM with more information about the setting. The *player knowledge* variable is the information provided to the player, and *world rules* is a set of additional world rules and information on the world. Both of these variables are unique for each story. The *STATE*, or state package, includes the *state transitions* variable, which notes down specific state transition conditions for each state, and the *state instruction* variable that tells the LLM how to handle the event states and the objective states in more detail. In this package, the *state transitions* variable is story-specific, and the *state instructions* variable is the same for every story. The *REASON*, or reason package, includes the *thinking* variable, which is an addition to the embedded CoT capabilities of the gpt-oss:120b model. The *memory instructions* variable is a prompt to tell the agent how to handle new information. Both of these variables are story independent and thus the same for each story setting. The *CHAR*, or character package, includes the *emotional state* variable, which tells the LLM how to react emotionally to how things change during the dialogue, and the *npc character* variable, which gives the LLM a more detailed persona to embody as the NPC. Both of these variables are specific to

the story setting. All the variable packages can be found in Appendix A.

For the variable packages used, it is expected that all of them should have an overall positive effect on dialogue consistency. However, depending on the variable, some have a stronger influence on consistency and others have a stronger influence on retention. World knowledge retention is dependent on the story setting as not every story setting has the same amount of world information to remember. It is also dependent on the variable package as some packages introduce extra world information. Context retention is also somewhat story setting dependent as not every story setting will allow for the same number of details to be shared. The variable packages could also have an influence on this as more information is provided to the NPC the character may behave differently.

3.4 Evaluating the Results

In order to evaluate dialogue consistency and narrative, the three story settings will be combined with the prompting packages. By doing this, we aim to determine the influence of the number of event states (few/many) and the number of objective states (few/many). As well as the influence of the different variable packages (positive/negative/neutral). For every story setting, we will use the *BASE* package, the *BASE* package plus any other package, and all packages together. Every story-variable package combination will be run 10 times, resulting in a total of 60 unique conversations for every story setting.

Dialogue consistency and narrative will be evaluated by measuring: *consistency*, *context retention*, and *world knowledge retention*, as was done in an earlier study [2]. *Consistency* will be measured by counting the total number of conflicting statements in a dialogue. This will give us a total number of consistency errors for each conversation. This will allow us to calculate the likelihood of an error for each story setting-variable package combination using Equation 1.

$$L = \frac{\sum E_{sv}}{\sum T_{sv}} \quad (1)$$

Here L is the likelihood of error, how often the error occurs each turn given the story setting-variable package combination. $\sum E_{sv}$ is the total number of consistency errors for a given story setting-variable package combination, and $\sum T_{sv}$ is the total number of turns given a story setting-variable package combination. For these, s indicates the story setting and v the variable package.

In order to measure *context retention*, the LLM will be asked to recall previously introduced information to the conversation. In order to measure *world knowledge retention*, the LLM will be asked to recall information about the world and story setting that was provided using a system prompt. After every 5 dialogue turns the LLM is provided with an evaluation prompt outside of the dialogue loop, until the objective state has been reached. This prompt asks the LLM to evaluate the dialogue consistency and provide a list with the *context retention* and *world knowledge retention*. This is evaluated manually according to Equation 2.

$$R = I - W - D \quad (2)$$

Here R is the retained information at a given dialogue turn, I is the total amount of retained information and W is the number of wrongly retained information. D is the amount of double retained information, that was already recalled at a previous evaluation turn. This means that the value of R can only increase as the evaluation turns go up. But the rate of increase is determined by the amount of newly retained information.

The cumulative context retention and world knowledge retention are given as the mean value across all dialogue sessions for every evaluation turn. To indicate the uncertainty of these estimates, a 95% confidence interval is calculated using the standard error of the mean, according to Equations 3–5. Although the same calculation is applied for every evaluation turn, the number of available dialogue sessions varies because not all conversations take an equal number of turns. Therefore, later evaluation turns are sometimes based on fewer observations, resulting in wider confidence bands.

$$SE = \frac{s}{\sqrt{n}} \quad (3)$$

Here SE is the standard error of the mean, s is the sample standard deviation of the cumulative retention values at a given evaluation turn, and n is the number of evaluated dialogue sessions.

$$CI = 1.96 \times SE \quad (4)$$

Here CI is the half-width of the 95% confidence interval. The factor 1.96 corresponds to the normal approximation for a 95% confidence interval. For small sample sizes, the resulting confidence intervals should be interpreted as approximate measures of uncertainty rather than exact inferential intervals.

$$[\bar{R} - CI, \bar{R} + CI] \quad (5)$$

Here $\bar{R}$ is the average cumulative retention at a given evaluation turn. With confidence bands both above and below this value.

After a conversation is done, the complete dialogue along with metadata is stored in both a `.txt` and a `.JSON` file. For every dialogue turn we store: the turn number, the user message, the NPC response, a thinking message that is not shown during dialogue, the word count of the response, the sentence count of the response, the time to first token (TTFT) and the total response time. For every evaluation turn we store: the turn number, the evaluation prompt, the raw response, the context retention and the world knowledge retention.

3.5 The NPC Dialogue Agent

Before starting any conversation, the story setting and variable packages are manually selected in the code. Then at first, a little information is printed to inform the user on the model that is used, what they need to do to stop the conversation and the *player knowledge* variable information is printed to let the user know about the story setting. Then the real dialogue loop starts. The user sends the first message, and then the LLM. After every five of these turns, the evaluation prompt is automatically sent to the LLM to question about *context retention* and *world knowledge retention*. A high-level overview of this process is displayed as a flowchart in Figure 1. An example of a complete conversation with the NPC dialogue agent can be found in Appendix B.

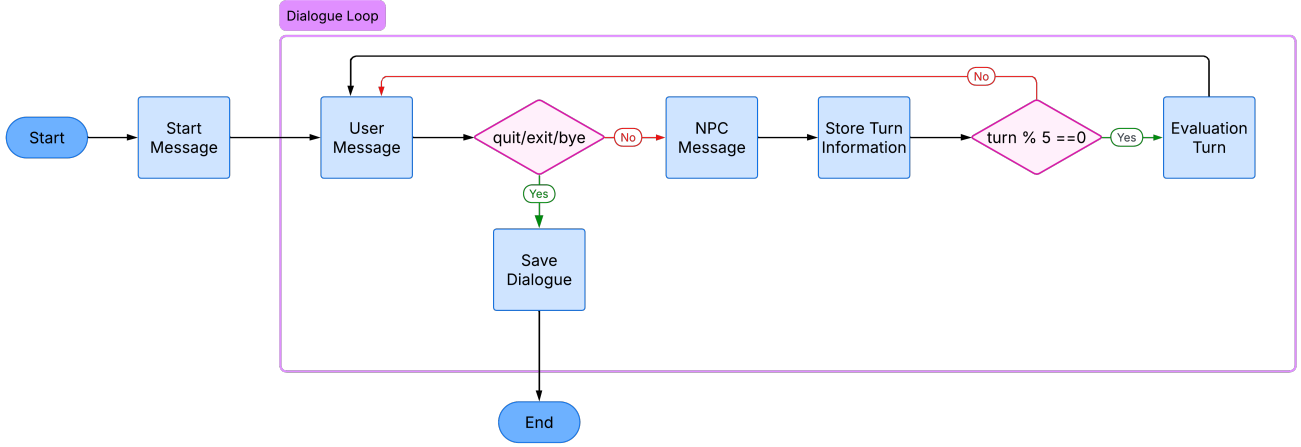


Figure 1: Flowchart of the NPC dialogue agent. This flowchart shows the process of having a conversation with the dialogue agent, from starting the conversation to ending it.

4 Results

In this chapter, the results of the experiments will be discussed. Both knowledge retention and consistency errors will be discussed to determine whether there are significant differences in dialogue quality and consistency among the three story settings, the variable packages used, and the number of event and objective states these stories have. An example of a full conversation with the NPC dialogue agent can be found in Appendix B. To evaluate the dialogue consistency, *context retention* and *world knowledge retention* were measured. *Context retention* was measured by asking the LLM to name all important information that was introduced in the dialogue. *World knowledge retention* was measured by asking the LLM to name all important information about the world and the conversation setting. For both of these calls, few-shot prompting was used, and it was asked after every five dialogue turns.

During the data generation and data analysis process, in addition to the standard *consistency error*, where an original statement is contradicted later in the conversation, five additional types of consistency errors were identified and these were also included in the analysis. All of these errors are a type of consistency error in the dialogue that would negatively impact immersion if a real NPC uttered them. They were all separately identified and counted according to Equation 1. These errors are: *formatting error*, here the output of the LLM was not in the desired format, but for example a list. The *simulating user* error is a type of error where the LLM answers as if it were the player. The *extra turn* error is when the LLM simulates multiple turns after one another as if the player had already answered. The *double message* error is when the LLM answers the player twice or more in one turn. The *extra turn* error and the *double message* error are very similar, as they both show an extra message in the conversation. A *thought bleeding* error happens when a part of the 'Thinking' process that the gpt-oss:120b module has gets included as a part of the response to the user. Examples for each error type for each story setting can be found in Appendix C.

In section 4.1 we will look at the comparison between the results of the three different story settings. In section 4.2 we will take a look at the results for the *FeFo* story setting, in 4.3 we will

take a look at the results of the *FeMo* story setting and in 4.4 we will take a look at the results from the *MeFo* story setting.

4.1 The Story Settings

In this section, we compare the results of the three different story settings with each other. First, we will compare the *context retention* and *world knowledge* retention, which is dependent on the story, in 4.1.1. Then we will compare the *consistency errors*, both between story settings and between variable packages in 4.1.2. More general metrics about the TTFT and message length is discussed in section 4.1.3.

4.1.1 Knowledge Retention

Figure 2 shows the comparison between the *context retention* of the three different story settings. Here, it is clearly visible that the average *context retention* for the *FeFo* story setting after several evaluation turns is lower than the other two settings. A reason for this could be that because there are fewer story event states to cover and objective states to reach, there are fewer contextual things to remember. The other two story settings with either many objective states or many event states are both outperforming this story setting, especially as the number of turns goes up. Another reason could be that this specific type of story setting, a police interrogation, inherently introduces fewer things to remember. Therefore, the LLM can remember fewer things overall, making this an unfair comparison. But in an ideal case, the LLM would at least remember one thing about every dialogue turn. Meaning all graphs should start at least at five and have a linear relation by remembering five new things every evaluation turn. The *MeFo* story setting matches this trend very well until evaluation turn 25, after which it slows down. The *FeMo* story setting performs well until evaluation turn 20, but the *FeFo* story setting already has a hard time after evaluation turn 10. As the number of turns goes up, we also see the confidence bands widen around the graphs. This is because not every conversation had an equal amount of dialogue turns, so where at evaluation turn 5 each of the three data points covers 60 unique conversations, at evaluation turn 30 this is much lower and differs per story setting.

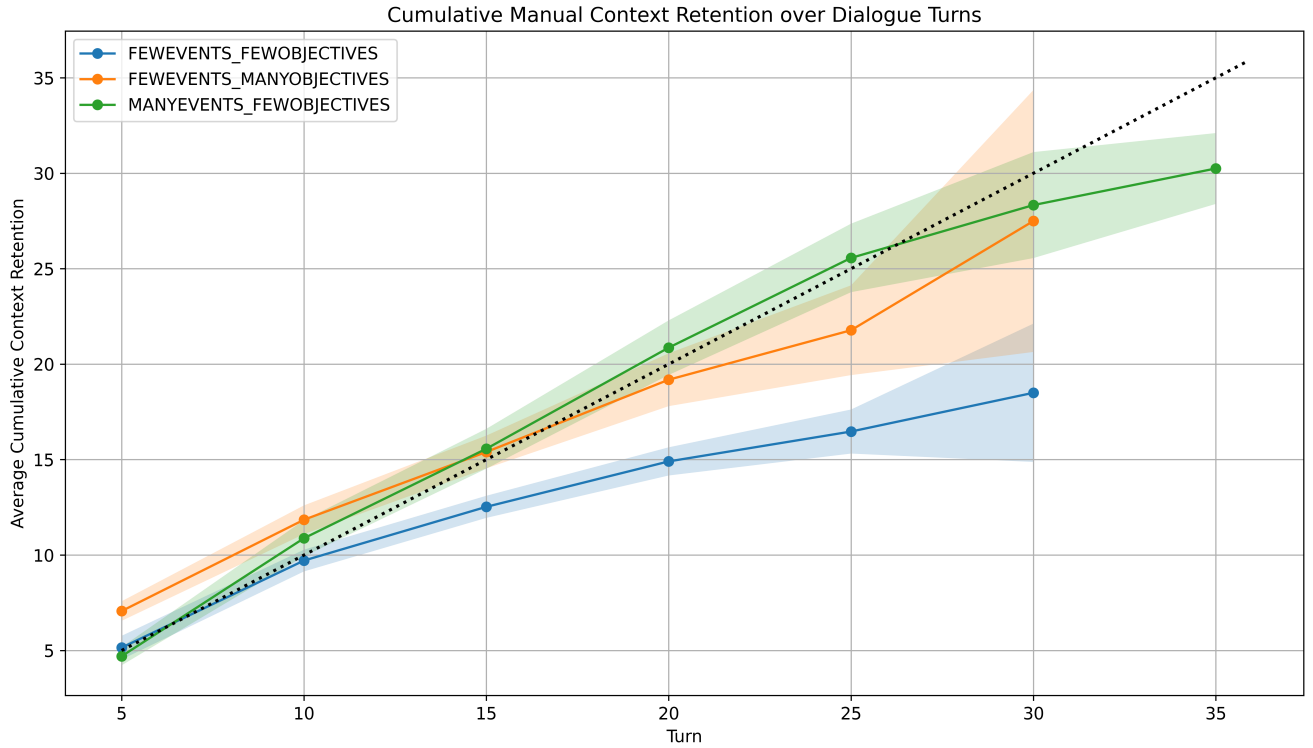


Figure 2: A comparison of context retention over time between the three different story settings. For this comparison, all conversations were included, 180 in total, 60 for every story setting. The shaded areas are the confidence bands. Because not every story ran for an equal number of turns, these widen in the later evaluation turns.

In Figure 3, the comparison between the *world knowledge retention* of the three story settings is shown. Here the graphs have swapped position, with now the *FeFo* story setting clearly outperforming the other two. The *MeFo* story setting also appears to slightly outperform the *FeMo* story on *world knowledge retention*. A possible explanation for this could be that as more information is introduced, it becomes more difficult for the LLM to accurately recall information about the setting of the story, as it is working harder on keeping track of the newly introduced information. However, there is also a maximum amount of world knowledge to remember. This maximum depends on both the story setting and on the variable packages that were included. Again making this comparison between story settings hard to draw any conclusions from. Still, in an ideal scenario, they would all be around their maximum at evaluation turn five and the only increase would be due to the different variable packages. However, we see that even after evaluation turn 20, the lines still go up. The retention number eventually goes down for the *FeMo* story setting. The reason for this, however, is the same reason that the confidence bands are so wide at the evaluation turn 30 data point. There were only two conversations in this setting that lasted this long, and they had a relatively low *world knowledge retention*.

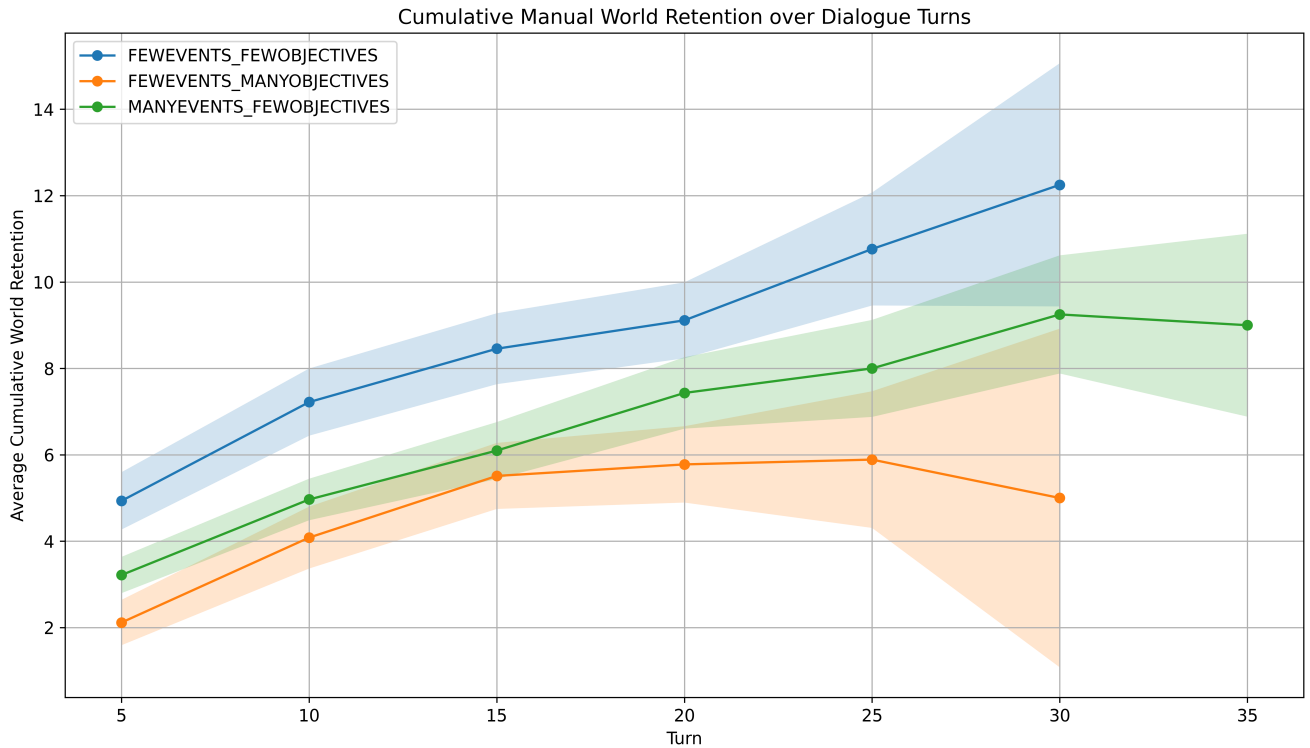


Figure 3: A comparison of world retention over time between the three different story settings. For this comparison, all conversations were included, 180 in total, 60 for every story setting. The shaded areas are the confidence bands. Because not every story ran for an equal number of turns, these widen in the later evaluation turns.

4.1.2 Consistency Errors

To measure dialogue consistency, *consistency errors* were counted at every turn, this was done manually. The results of this analysis are shown in Figure 4. The number of *consistency errors* appears to be relatively consistent across the three story settings, with the number of errors in the *FeMo* setting being only slightly higher than the other two. Another find here is the large number of *extra turn errors* that were found in the *MeFo* story setting. This could be the case because a lot more information about upcoming events is already given, so the LLM is more likely to already start talking about it before its current turn is over. This would suggest that there still is a need for a separate state tracking module for LLM-generated dialogue, to prevent the LLM from continuing a conversation out of turn.

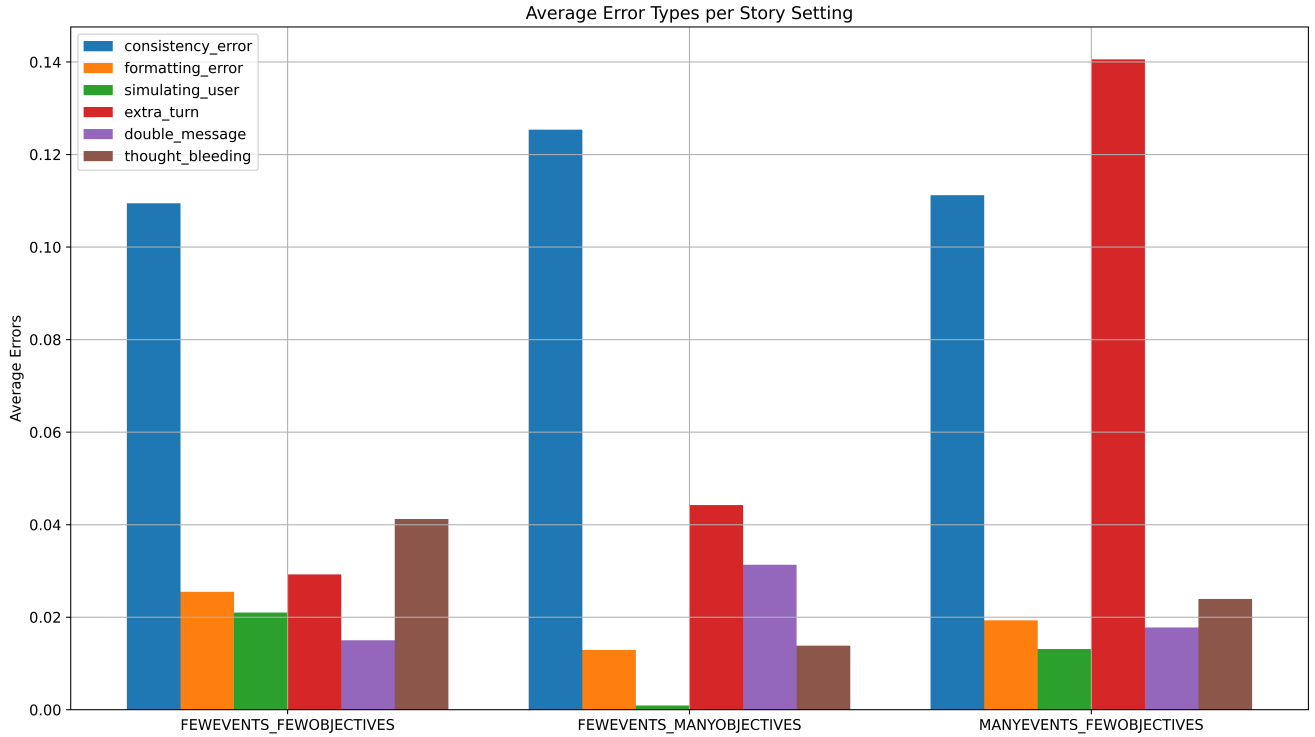


Figure 4: A comparison of the consistency error and the other error types that were found between the three story settings. For this comparison, all conversations were included, 180 in total, 60 for every story setting.

We can also look at the error rate depending on the variable package instead of the story setting, this is done in Figure 5. Here we can see that most variable packages show a similar image. The *consistency error* is the highest, followed by the *extra turn* error, and the remaining errors are significantly lower than these two. However, looking at Figure 4, we see that the *extra turn* error is most prevalent in the *MeFo* story setting. If we then look at the heatmap in Figure 6, where we can see the likelihood for every error type given a story setting and variable package. We see that most of the likelihood of a *extra turn* error comes from the *MeFo* story setting with the *REASON* variable package. The *KNOW*, *STATE* and *CHAR* variable packages also significantly contribute to a high likelihood of *extra turn* errors. This means that the height of the *extra turn error* bar in Figure 5 is somewhat inflated because of these high values in this particular variable package-story setting combination. Furthermore, we can see in Figure 5 that the *CHAR* variable package has an overall relatively high error rate, with all except for the *simulating user* error being the most likely compared to the other packages. This could suggest that more details about the character the NPC needs to embody will lead to more consistency errors.

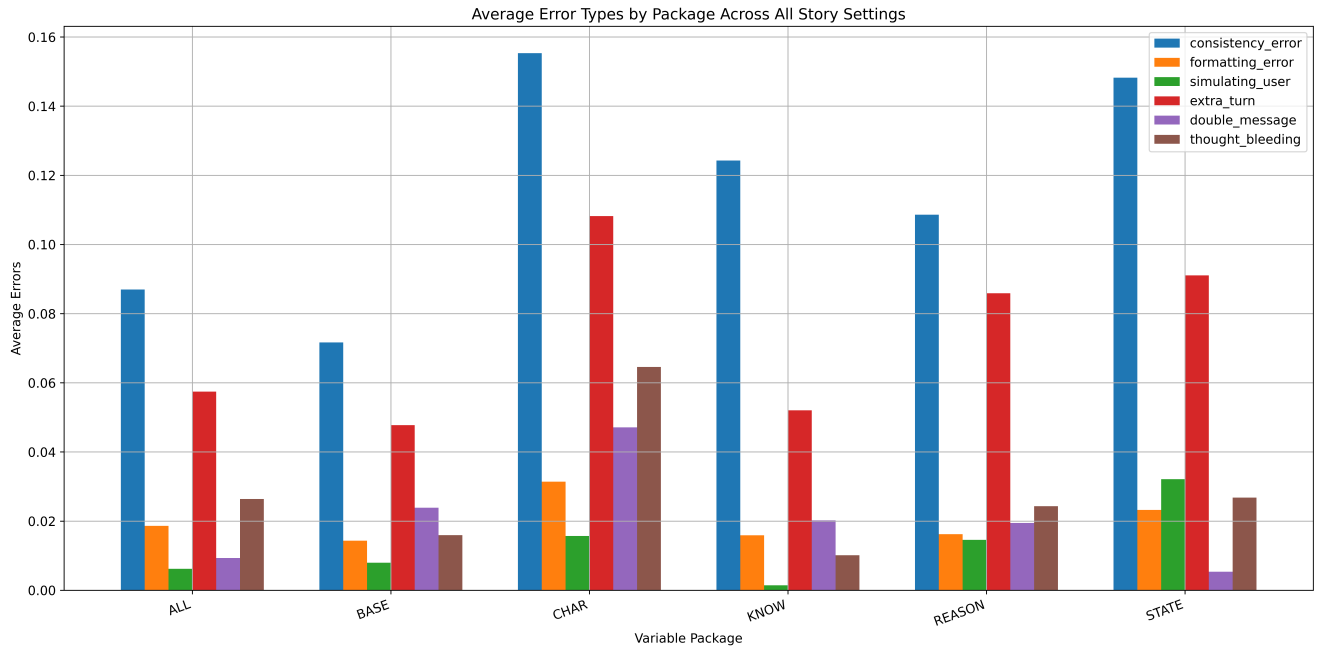


Figure 5: A comparison between the consistency error and other error types that were found between the six different variable packages. For this comparison, all conversations were included and combined, which means 30 for every variable package.

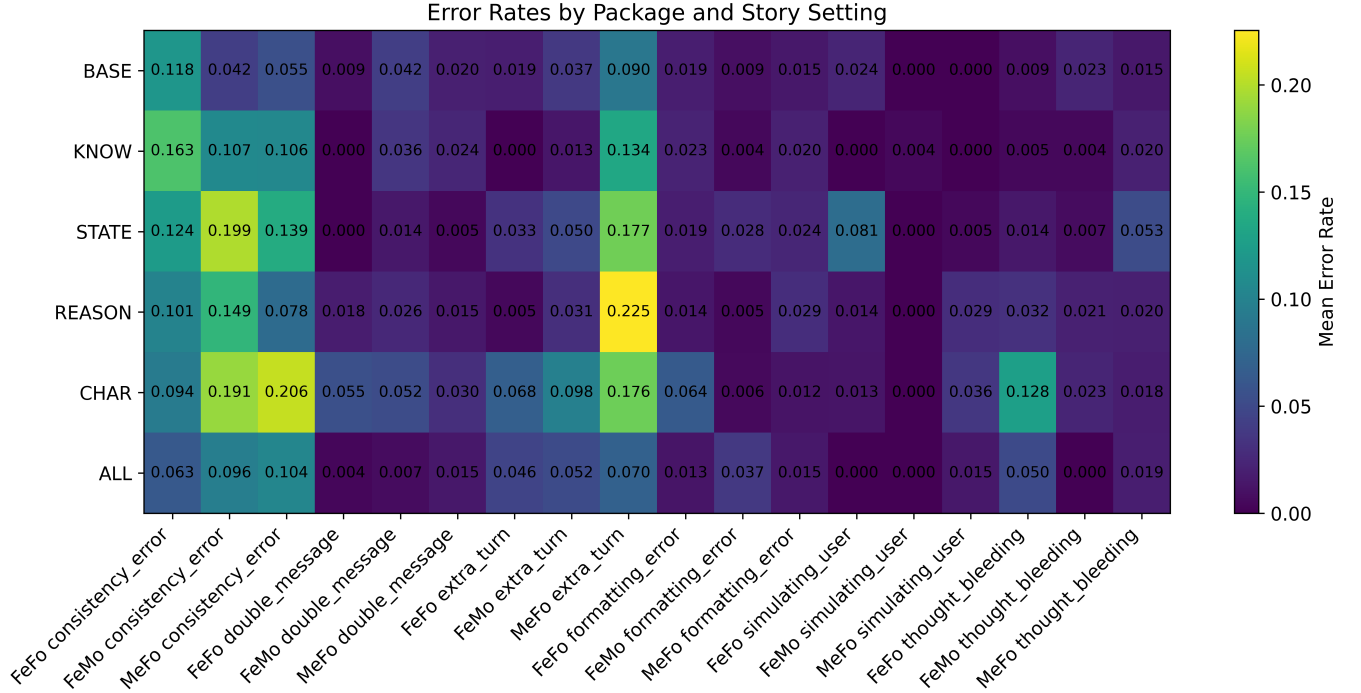


Figure 6: A heatmap comparing the consistency error and other error types that were found between the three story settings and the variable packages. For this comparison, all conversations were included, 60 for every story setting, 30 for every package, 180 in total. The lighter the square, the more probable the error-variable-story combination.

Looking at Figures 3 and 4 it appears that adding extra event states to a story has negative effects on story consistency, however this does not tell the full picture, as in these figures no distinction is made between the different variable settings. Figures 5 and 6 provide a better overview of the errors depending on the variable package. But a better overview of the difference within the story setting will be discussed in the remainder of this chapter.

4.1.3 General Metrics

In this section, we will look at some general story metrics. Figure 7 shows a comparison between the three different story settings. We see the TTFT and the total time it took the LLM to complete a message in seconds. For every story setting the average TTFT was at least 1.5 seconds, but for the *MeFo* story setting it was more than 2.5 seconds. This does not sound like a lot, but during gameplay these waiting times can limit immersion. Definitely, if the total response time is over 3.5 seconds. This suggests that although not terrible, response times could be better, and that on an average PC this value could be even higher.

For all three story setting the prompts asked for 3 sentences in the response. This is the case. But for the *MeFo* story setting all three metrics are a bit higher this could be because there are more event states and thus the LLM has more information to go over and to response to. Resulting in both longer waiting times as longer responses.

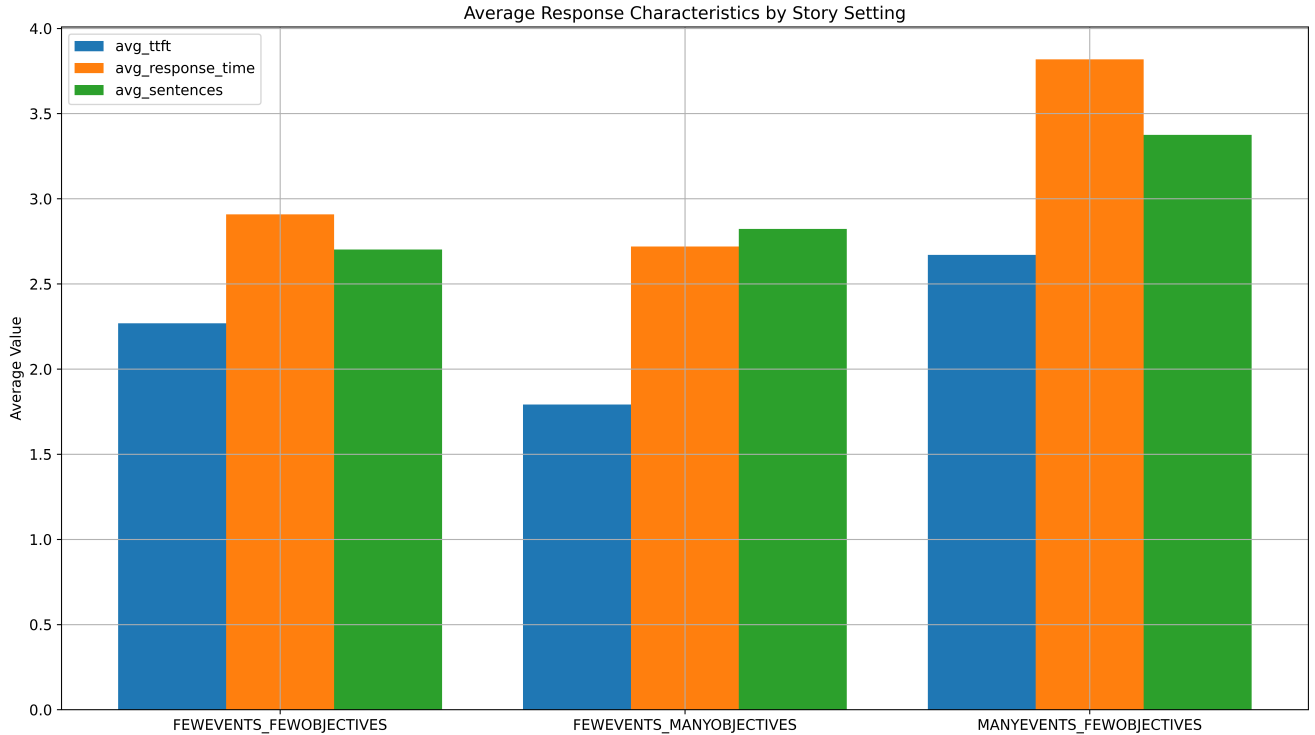


Figure 7: A comparison between three general metrics of the story settings. Here, avg_ttft and avg_response_time are in seconds, and avg_sentences is in number of sentences. For this graph, all dialogue was included, so 60 conversations per story setting.

4.2 Few Events - Few Objectives

This section is about the results from the *FeFo* story setting, a police interrogation story setting with only two event states and two objective states. As can be seen in Figures 2 and 3, for this story setting the *context retention* was relatively low and the *world knowledge retention* was relatively high compared to the other two story settings. From Figure 4 we know that the error rates were comparable to those of the other two story settings, but with a relatively low *extra turn* error and a relatively high *thought bleeding* error rate. In this section, we will look at the influence of the variable packages on these metrics.

4.2.1 Knowledge Retention

Figures 8 and 9 show the average number of *context retentions* and *world retentions* after a number of turns for the six different variable settings. The first thing that is clear is that not every variable setting had an equally long number of dialogue turns. Also, because none of the variable combinations have more than half of their conversations reaching beyond 20 turns, we will not read too much into what happens after evaluation turn 20.

But if we look at the first 20 turns in Figure 8, it looks like the *BASE*, *KNOW* and *CHAR* variable combinations are slightly underperforming compared to the *STATE*, *REASON* and *ALL*.

This is interesting because the *CHAR* and the *KNOW* variable packages both add extra information about the story setting, whereas the *REASON* and *STATE* are more about the pace of the story and how to reason. This suggests that more information about the story setting does not improve the context retention within the dialogue, but rules about formatting and reason do. Until evaluation turn 10 the *CHAR*, *ALL*, *REASON* and *STATE* packages follow the trend of one remembered thing for every turn, but none of them managed to continue this trend to evaluation turn 15 or beyond. Suggesting that it was hard for the LLM to find context to remember. Seeing as all packages struggle to follow this minimum, this could be story-specific.

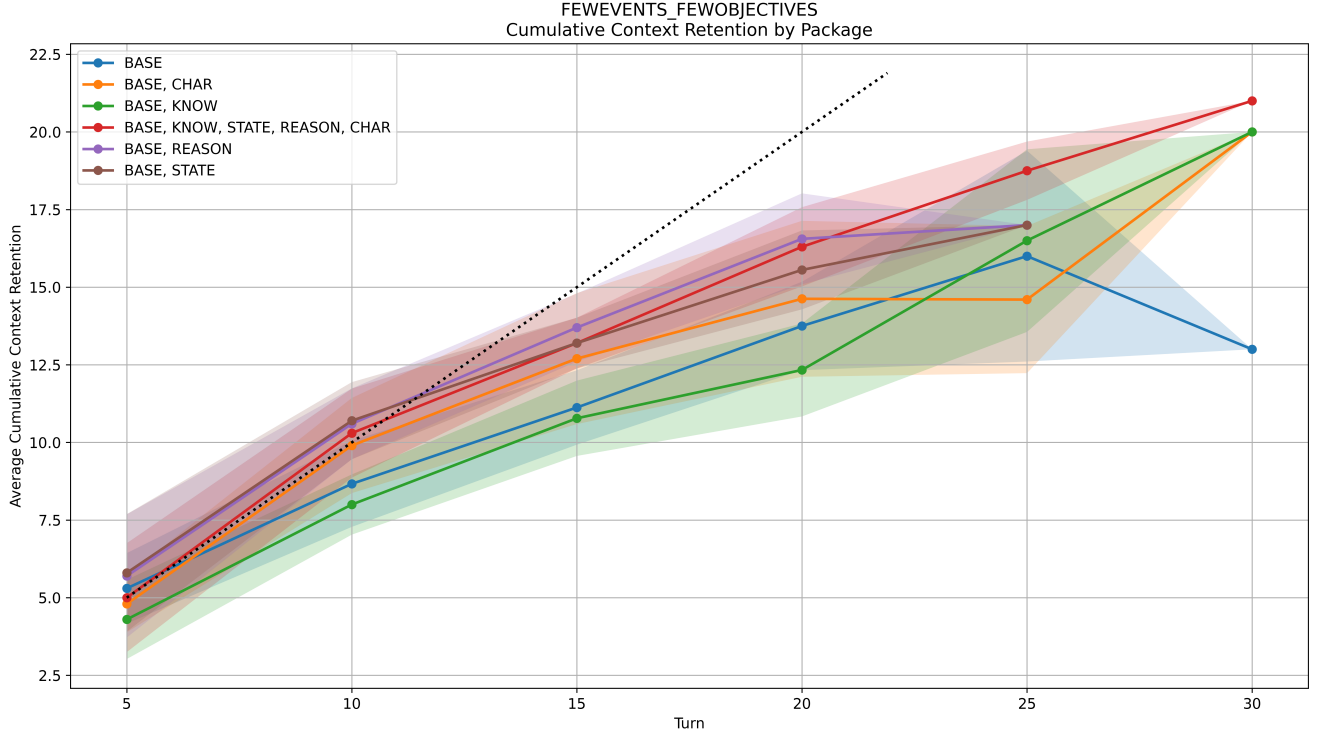


Figure 8: A comparison of context retention over time between the six different variable packages for the *FeFo* story setting. For this comparison, all conversations of the setting were included, 60 in total, 10 for every variable setting. The shaded areas are the confidence bands, as not every story ran for an equal number of turns, these widen in the later evaluation turns. When the bands narrow to a single point, it means there is only data for a single conversation for that data point.

Now looking at Figure 9, again only looking at the first 20 turns, we see the roles reversed. Here, the *STATE* and *REASON* variable packages perform comparably to the *BASE* package. Whereas the *KNOW* and the *CHAR* packages are able to achieve higher world knowledge retention. A possible explanation for this could be that with these packages there is just more information to remember, and so they achieve higher average world knowledge retention. However, if this is the case, the *ALL* variable package does not appear to be negatively impacted by the *REASON* and *STATE* packages. It seems like it just combines the extra information from the *CHAR* and *KNOW* packages. So it appears that the *KNOW* and *CHAR* packages have a positive effect, whereas the *REASON* and *STATE* packages have a neutral influence on the world retention performance.

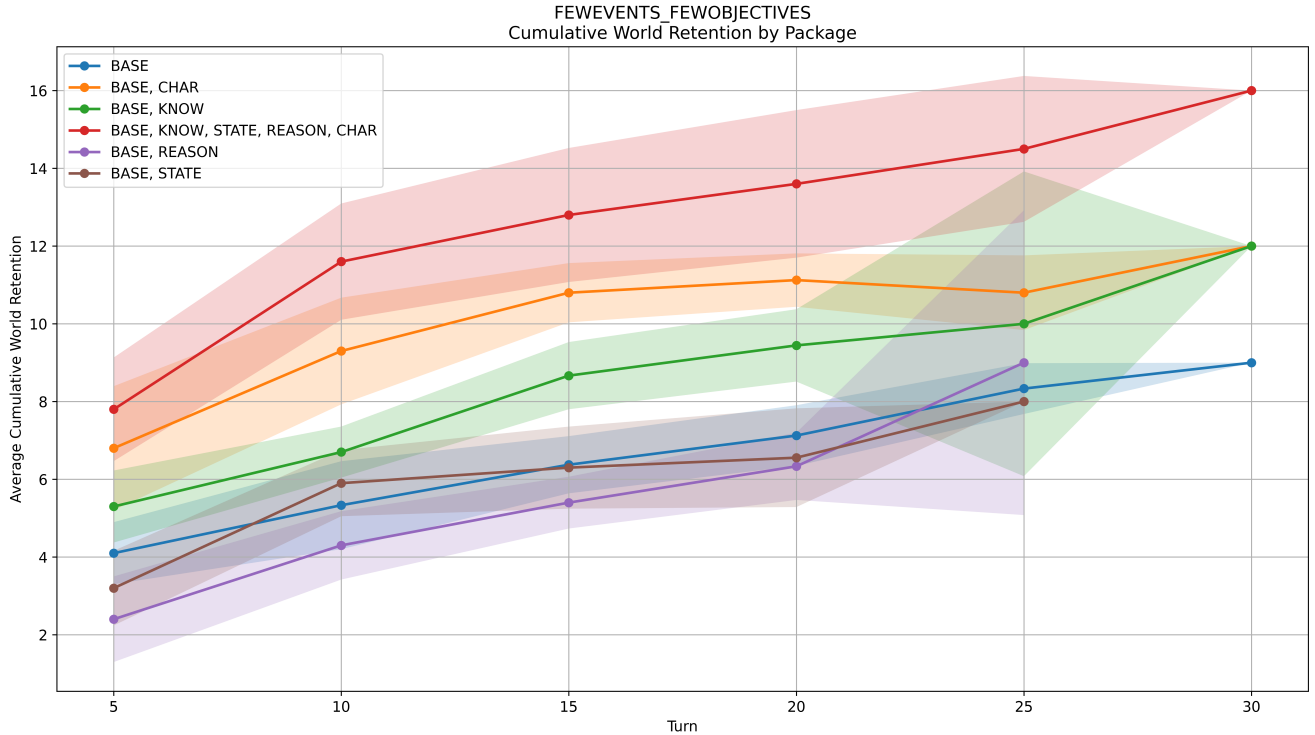


Figure 9: A comparison of world knowledge retention over time between the six different variable packages for the *FeFo* story setting. For this comparison, all conversations of the setting were included, 60 in total, 10 for every variable setting. The shaded areas are the confidence bands, as not every story ran for an equal number of turns, these widen in the later evaluation turns. When the bands narrow to a single point, it means there is only data for a single conversation for that data point.

4.2.2 Consistency Errors

In Figure 10 the different types of errors for the different types of variable packages in the *FeFo* story setting are shown. The *BASE* package has all errors however, the *consistency error* is significantly more present. The *CHAR* package has the highest chance of errors, however consistency is not the highest. The *thought bleeding* error has the highest error percentage, and *formatting error*, *extra turn* and *double message* are also higher than in any other package. The *KNOW* package has a very low chance at most errors, but the *consistency error* chance is here the highest out of all errors across the packages. The *REASON* package has a very similar profile to that of the *BASE* package. With the *consistency error* by far the highest and the other all relatively low. Both the *KNOW* and *REASON* packages seem to improve the *simulating user* and *extra turn* errors compared to *BASE*. For *KNOW* an improvement is also made for the *double message* error. This could be because these two packages provide rules about the world and how to reason, lowering out-of-turn errors as a result. For the *STATE* package the big outlier is the high *simulating user* error, which is by far the highest among all packages. A reason for this could be that because the LLM knows more about the states, it starts to continue the conversation with itself, as it already

knows where the story is going. The *ALL* package has the lowest *consistency error* chance out of all packages, however the *extra turn error* and the *thought bleeding error* are relatively high.

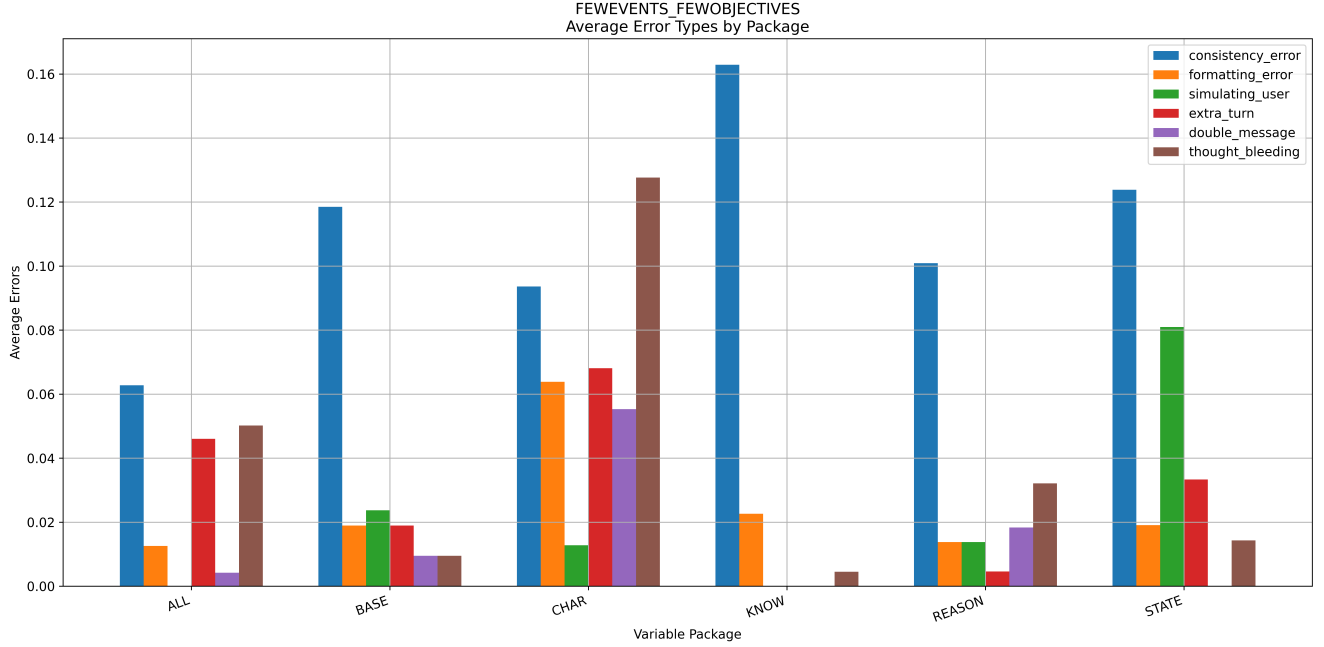


Figure 10: A comparison between the error types that were found between the six different variable packages for the *FeFo* story setting. For this comparison, all 60 conversations of the story setting are included, 10 for every variable package.

Figure 11 shows a heatmap of the errors in the *FeFo* story setting. It shows the likelihood that an error is gonna occur more than a given number of times. This way we can see that some errors occur often, but mostly once per conversation. While other errors occur multiple times in a single conversation. The *consistency error* is the most likely to occur, but for the knowledge package, it occurs more often in a single conversation. The *double message* error on the other hand happens a lot with the *CHAR* variable package, but most often only once. The *thought bleeding error*.

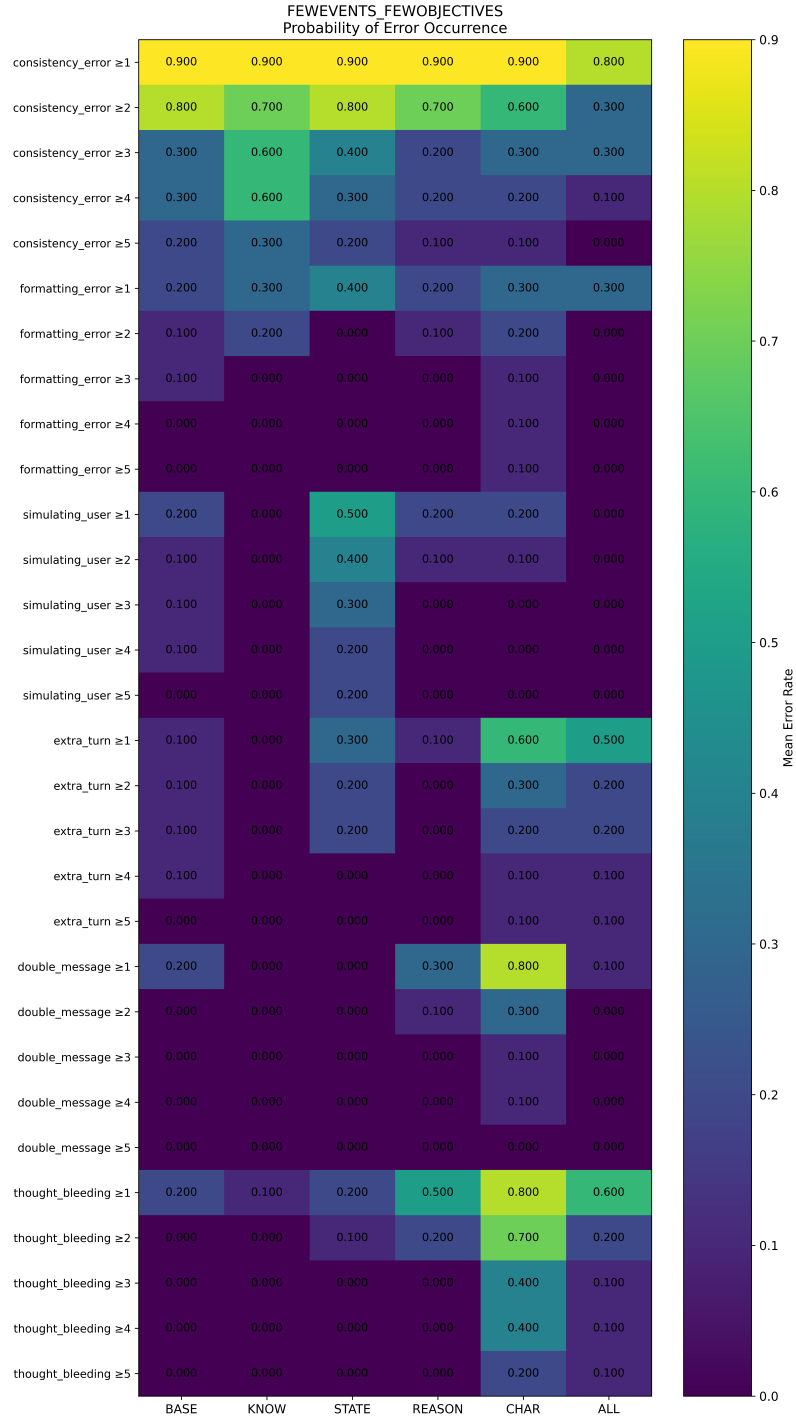


Figure 11: A heatmap comparing the consistency error and other error types that were found in the *FeFo* settings and the variable packages. This shows the likelihood that an error will occur more than a given number of times if it occurs in a single conversation. For this comparison, all 60 conversations were included, 10 for every variable package. The lighter the square, the more probable the error-variable-story combination.

4.3 Few Events - Many Objectives

In this section, we will look at the results from the *FeMo* story setting, a setting about escaping a fallen nation. As was seen in Figures 2 and 3, the *context retention* was comparable to that of the *MeFo* story setting, whereas the *world knowledge retention* was relatively low compared to the other two story settings. From Figure 4 we know that, the likelihood of a *consistency error* was the highest for this story setting. Whereas the likelihood of a *simulating user* error was the lowest of any type of error in any story setting. We will have a closer look at the influence of the variable packages on this in this section.

4.3.1 Knowledge Retention

The average conversation length for this story setting was lower than 20, so we will focus our attention on the first 3 evaluation turns. Looking at Figure 12, we see a somewhat similar image to Figure 8, with all graphs relatively close together. Also, again with the *CHAR* and *KNOW* packages performing similarly to *BASE*. However, the *REASON* package now also performs closer to *BASE*, with the *STATE* package outperforming the *ALL* package. After evaluation turn 15, the data starts spreading more, this is due to the fact that there are very few conversations for these data points. But like for the *FeFo* story setting, it appears that introducing more information about the story setting here also does not have a positive influence on the context retention. However, all graphs do fall within the confidence bands of at least two other graphs, so with more data shifts are possible.

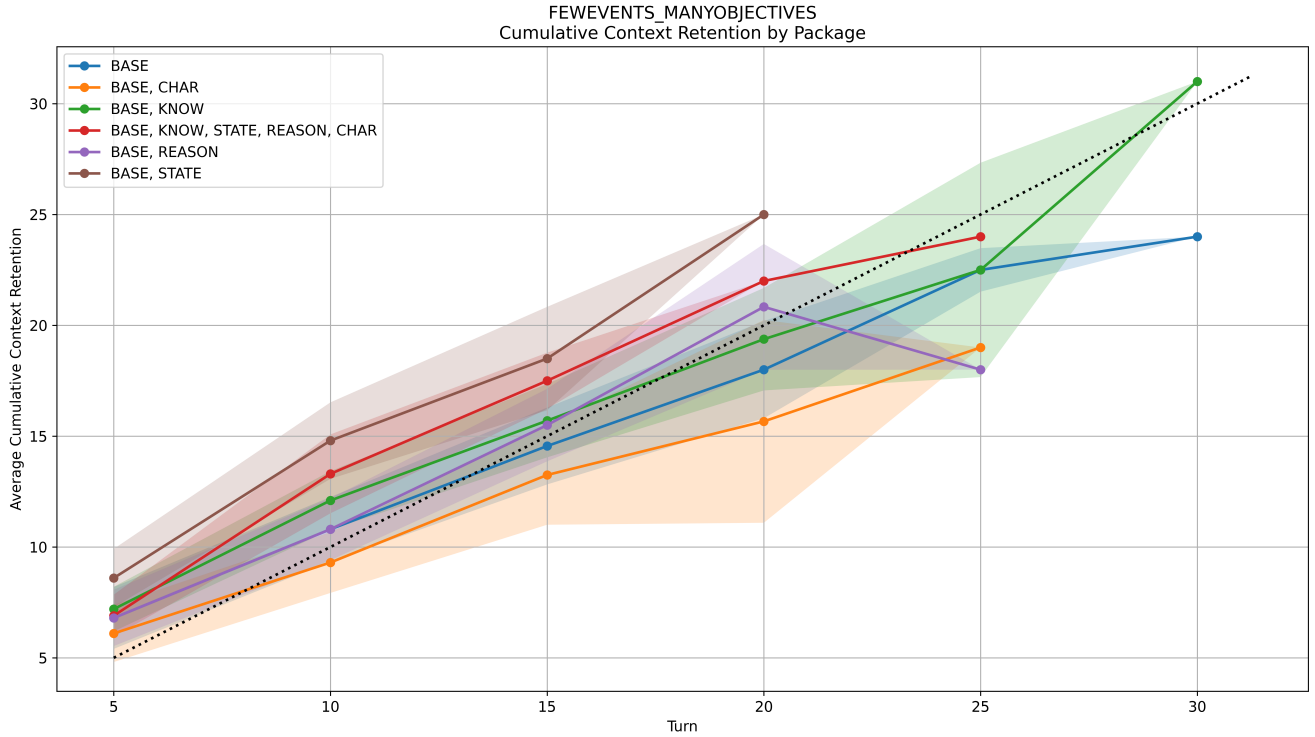


Figure 12: A comparison of context retention over time between the six different variable packages for the *few events many objectives* story setting. For this comparison, all conversations of the setting were included, 60 in total, 10 for every variable setting. The shaded areas are the confidence bands, as not every story ran for an equal number of turns, these widen in the later evaluation turns. When the bands narrow to a single point, it means there is only data for a single conversation for that data point.

The comparison for *world knowledge retention* for this story setting is shown in Figure 13. Here we see that the *ALL* package does not necessarily guarantee high results, as in this case it performs very close to *BASE*, along with the *REASON* package, and the *STATE* package performs worse than *BASE*. The *KNOW* package only slightly outperforms *BASE* and only the *CHAR* package stands clear of the others. It appears that here, providing more information about the story setting does not increase *world knowledge retention* by that much, and that providing other information, like about the event states, could limit the LLM’s ability to remember information about the story setting. And where for the *FeFo* story setting the *ALL* package was a combination of the improvements, here the inclusion of the *CHAR* package does not appear to outweigh any limiting influence on the *ALL* package.

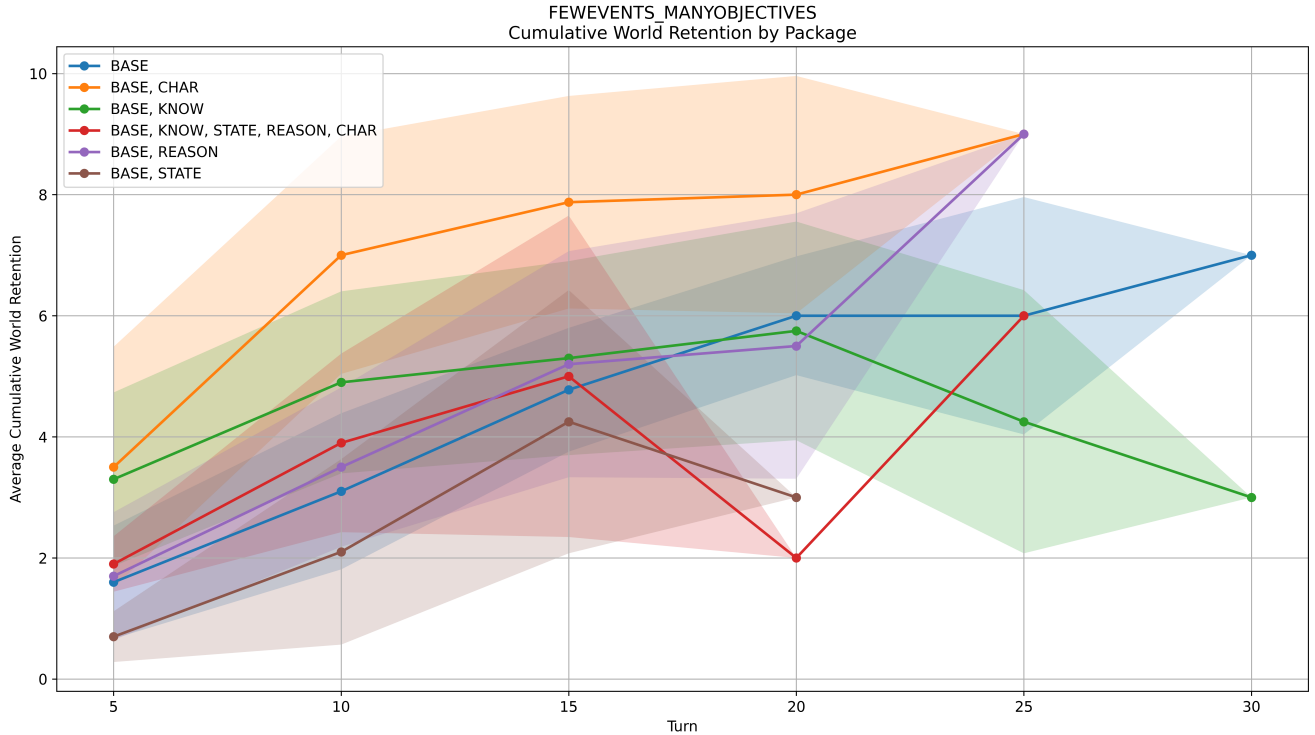


Figure 13: A comparison of world knowledge retention over time between the six different variable packages for the *FeMo* story setting. For this comparison, all conversations of the setting were included, 60 in total, 10 for every variable setting. The shaded areas are the confidence bands, as not every story ran for an equal number of turns, these widen in the later evaluation turns. When the bands narrow to a single point, it means there is only data for a single conversation for that data point.

4.3.2 Consistency Errors

Looking at the errors for this story setting in Figure 14, we also see that the *consistency error* is the most prominent error in this in these conversations. We see that the *BASE* package has relatively low error rates and as more information is introduced, the error rates increase. In the *CHAR* package, we see an increase in the *consistency error* and the *extra turn error*. For the *KNOW* package, almost all error rates drop, except for the *consistency error* and it is the only package with a recorded *simulating user error*. In the *REASON* package, all errors drop a little in likelihood, again except for the *consistency error*. This is a similar pattern to what was seen for the *FeFo* story setting. In the *STATE* package, some values drop a little and some rise a little, but the *consistency error* again increases by a lot. The *ALL* package is next to a combination of all packages a good average of the packages, as compared to *BASE*. Compared to this package, the *formatting error* and *extra turn error* increase in likelihood, the *double message* and *thought bleeding* errors decrease in likelihood, and the *consistency error* increases, but it increases by less than it does for the individual packages.

The fact that only one package had a *simulating user error* suggests that this is probably

something story-specific, the other two story settings have this error more often. What is also interesting is that even though the *formatting error* likelihood is low among all variable packages, the relatively higher value in the *STATE* package appears to be the reason for the higher value of the error in the *ALL* package.

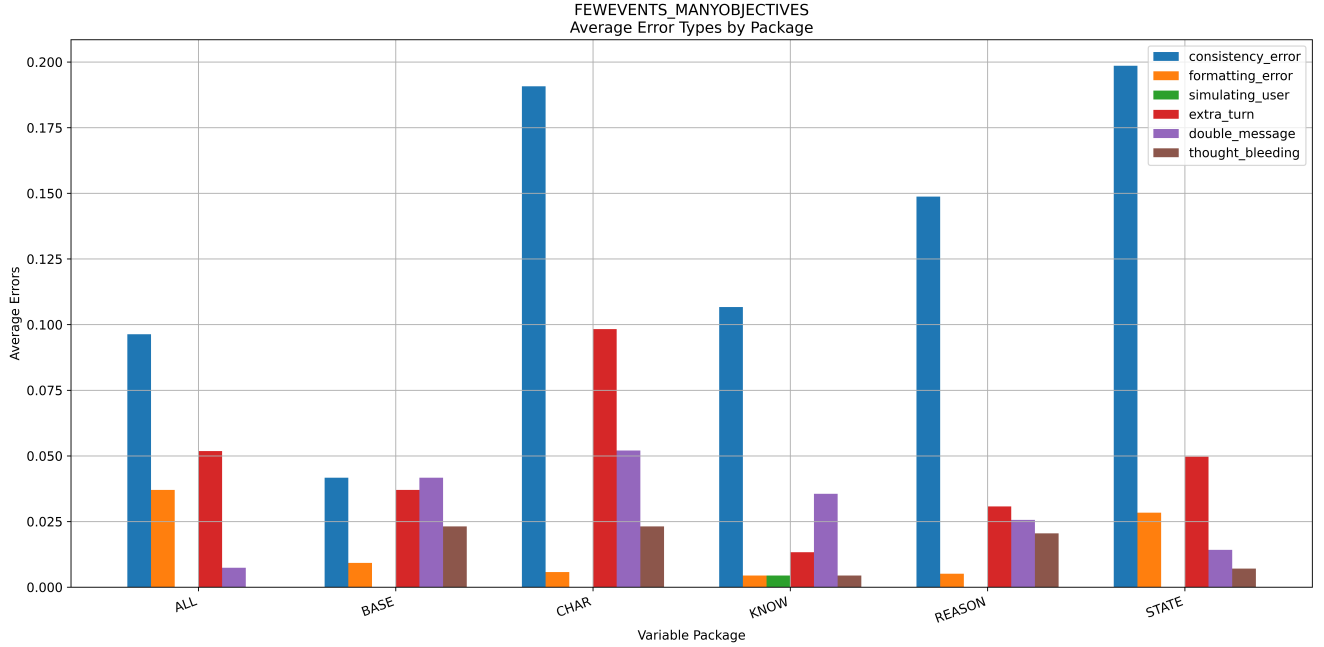


Figure 14: A comparison between the error types that were found between the six different variable packages for the *FeMo* story setting. For this comparison, all 60 conversations of the story setting are included, 10 for every variable package.

In the heatmap for this story setting, shown in Figure 15, we can see that the *formatting error* and *simulating user error* almost do not occur. We also see that for the *state* and *ALL* package, errors usually only occur once in a conversation. But in the *BASE* and *CHAR* packages, errors often happen more than once. Especially the *extra turn error* and the *double message error* often happen more than once. The *consistency error* is also the most prevalent here and often happens more than once per conversation.

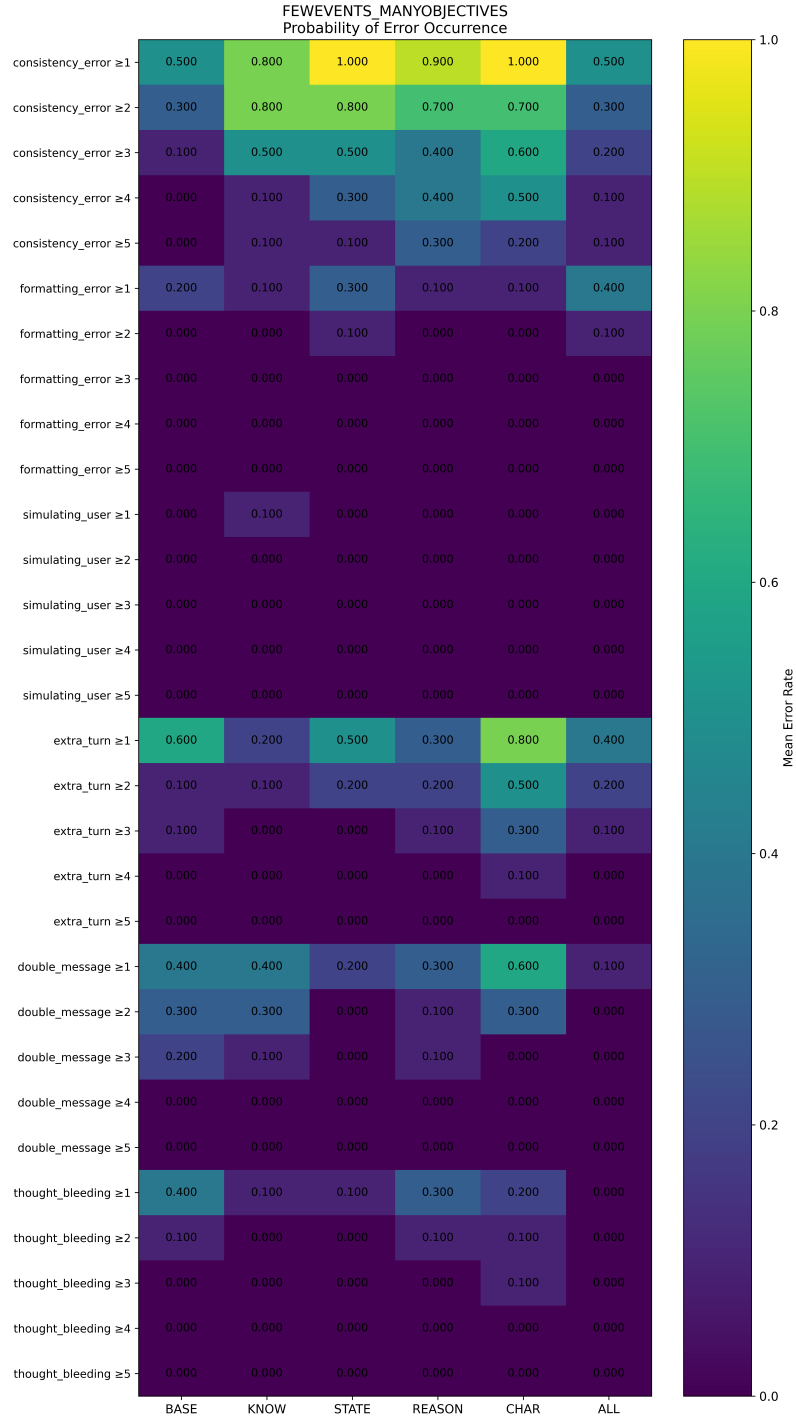


Figure 15: A heatmap comparing the consistency error and other error types that were found in the *FeMo* settings and the variable packages. This shows the likelihood that an error will occur more than a given number of times if it occurs in a single conversation. For this comparison, all 60 conversations were included, 10 for every variable package. The lighter the square, the more probable the error-variable-story combination.

4.4 Many Events - Few Objectives

This section looks at the results of the *MeFo* story setting, this setting was a dialogue between two astronauts doing repairs on the outside of a spaceship. This story setting had a marginal edge in *context retention* and performed average in *world knowledge retention* compared to the other two story settings. In terms of *consistency errors*, the most significant difference compared to the other two story settings was the high likelihood of *extra turn* errors. In this section, we will look at what influence the different variable packages had on these things.

4.4.1 Knowledge Retention

The average dialogue length for this story setting was less than 25 turns, so we will mostly focus on the results up until turn 20 for the retention graphs. Figure 16 shows the average context retention during the dialogue of this conversation setting. Like for the other two conversation settings, the graphs of the variable packages are relatively close together. The *BASE* package does appear to have the worst performance, together with the *KNOW* package. However, the *CHAR*, *REASON* and *ALL* packages are not significantly better, all falling within the confidence interval. Even the *STATE* package only really improves after 15 turns, and even that is marginal. Most variable packages do make the minimal trend of remembering at least one thing per turn on average. Suggesting that this story setting with more event states does allow for more *context retention*.

After 20 turns, the *CHAR* package value shoots up. But this is due to a single outlier data point that pulls it high above the rest and is not an accurate reflection, as proven by the data points in previous evaluation turns. Another thing to note is that the conversation length was quite variable for this story setting. Depending on the variable package, the difference between the longest conversation length and the shortest conversation length could be more than 20 turns.

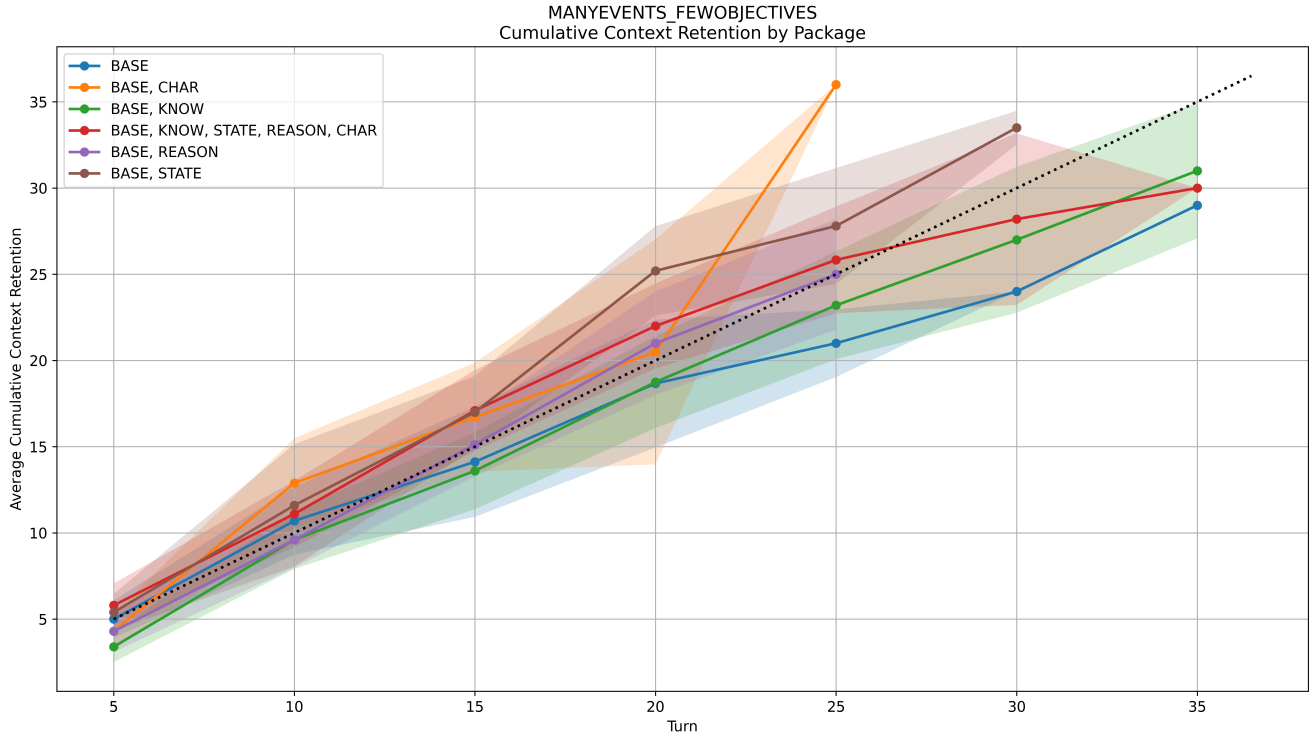


Figure 16: A comparison of context retention over time between the six different variable packages for the *MeFo* story setting. For this comparison, all conversations of the setting were included, 60 in total, 10 for every variable setting. The shaded areas are the confidence bands, as not every story ran for an equal number of turns, these widen in the later evaluation turns. When the bands narrow to a single point, it means there is only data for a single conversation for that data point.

Looking at the world retention in Figure 17, we see a much clearer separation between variable packages that do improve the *world retention* and packages that have little to no influence on the amount of story setting information that is remembered. The *REASON* and *STATE* packages perform similarly to the *BASE* package and the *CHAR* package only shows an improvement on these after 15 turns. The rapid increase to 14 things remembered comes from one single conversation and is thus not an accurate reflection. But it does appear that with this package the LLM remembered information about the world a bit better later in the dialogue. The *KNOW* and *ALL* packages are clearly outperforming the others. Where the performance from the *ALL* package likely comes from the performance of the *CHAR* package that is included in it, as it matches its performance closely. The *ALL* package only outperforms the others a little after turn 15, which would be because of the *CHAR* package that appears to trigger *world retention* a bit later on in the dialogue. Other than that, the improvement in *world knowledge retention* appears to come solely from the *KNOW* package.

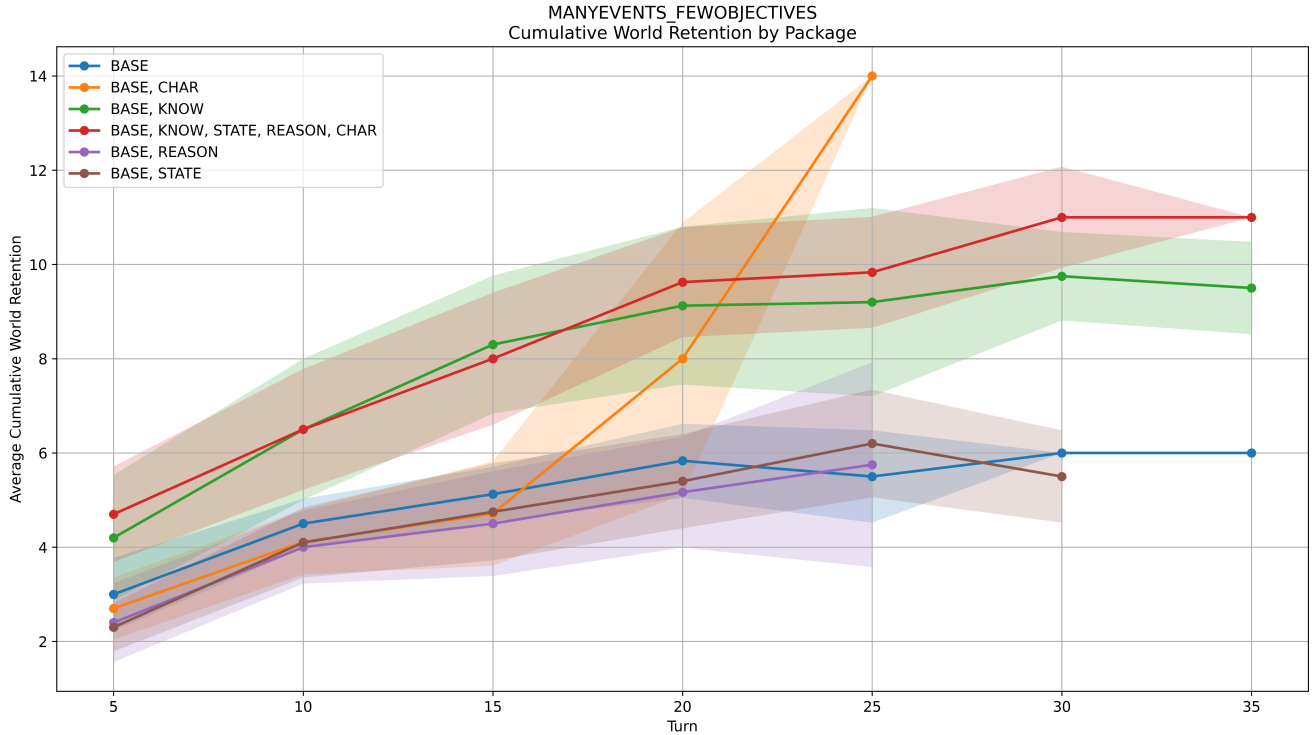


Figure 17: A comparison of *world knowledge retention* over time between the six different variable packages for the *MeFo* story setting. For this comparison, all conversations of the setting were included, 60 in total, 10 for every variable setting. The shaded areas are the confidence bands, as not every story ran for an equal number of turns, these widen in the later evaluation turns. When the bands narrow to a single point, it means there is only data for a single conversation for that data point.

4.4.2 Consistency Errors

Looking at Figure 18, we can see why in Figures 4 and 5 the *extra turn error* was so high. Whereas in the other two story settings this type of error often had the second or third highest chance to happen for a given package, for this story setting it always has the highest or second highest chance. In the *BASE*, *KNOW*, *REASON* and *STATE* packages, the *extra turn error* is the highest. As opposed to the *FeFo* and *FeMo* story settings, here the *KNOW* and *REASON* packages are not really improvements on the *BASE* variable package. The *KNOW* package shows a similar profile, but all error rates except for *simulating user* increase a little. For the *REASON* package, the increase, especially in the *extra turn error*, is even more significant. The likelihoods of errors in the *ALL* package are relatively low and comparable to those of the *BASE* package. Except for the *consistency error*, which is almost twice as likely, and the *simulating user* error, which did not appear in the *BASE* package. It appears that the many event states in this story setting cause there to be more *extra turn* errors. Especially in the *REASON*, *CHAR*, *STATE* and *KNOW* variable packages. It could be that because there are so many more states, with more information about the story, the LLM continues the story without waiting on the user, as it already knows how

the story goes.

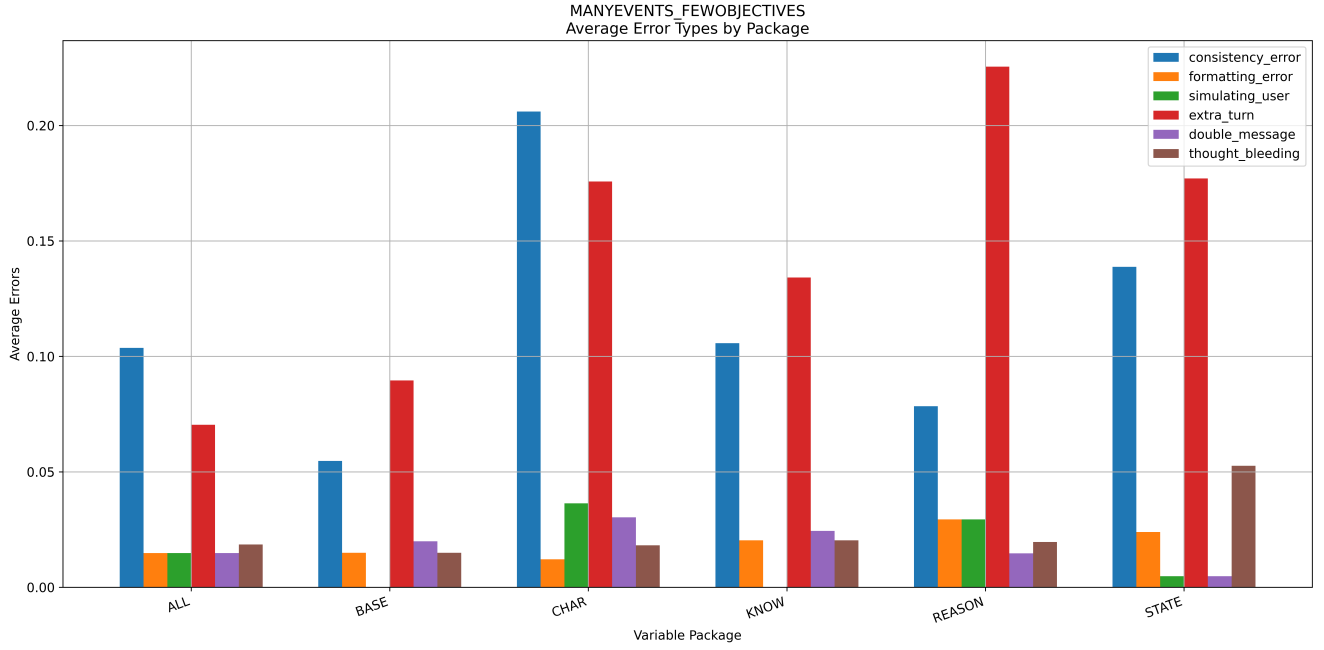


Figure 18: A comparison between the error types that were found between the six different variable packages for the *few events many objectives* story setting. For this comparison, all 60 conversations of the story setting are included, 10 for every variable package.

The heatmap for the *MeFo* story setting is shown in Figure 19. Here we can clearly see that the *consistency error* together with the *extra turn error* is the most prevalent. However, especially the *extra turn error* often happens more than once, for the *REASON* variable package, even often more than 5 times. The *thought bleeding error*, on the other hand, usually only happens once, except for the *STATE* package, where it also happens more often.

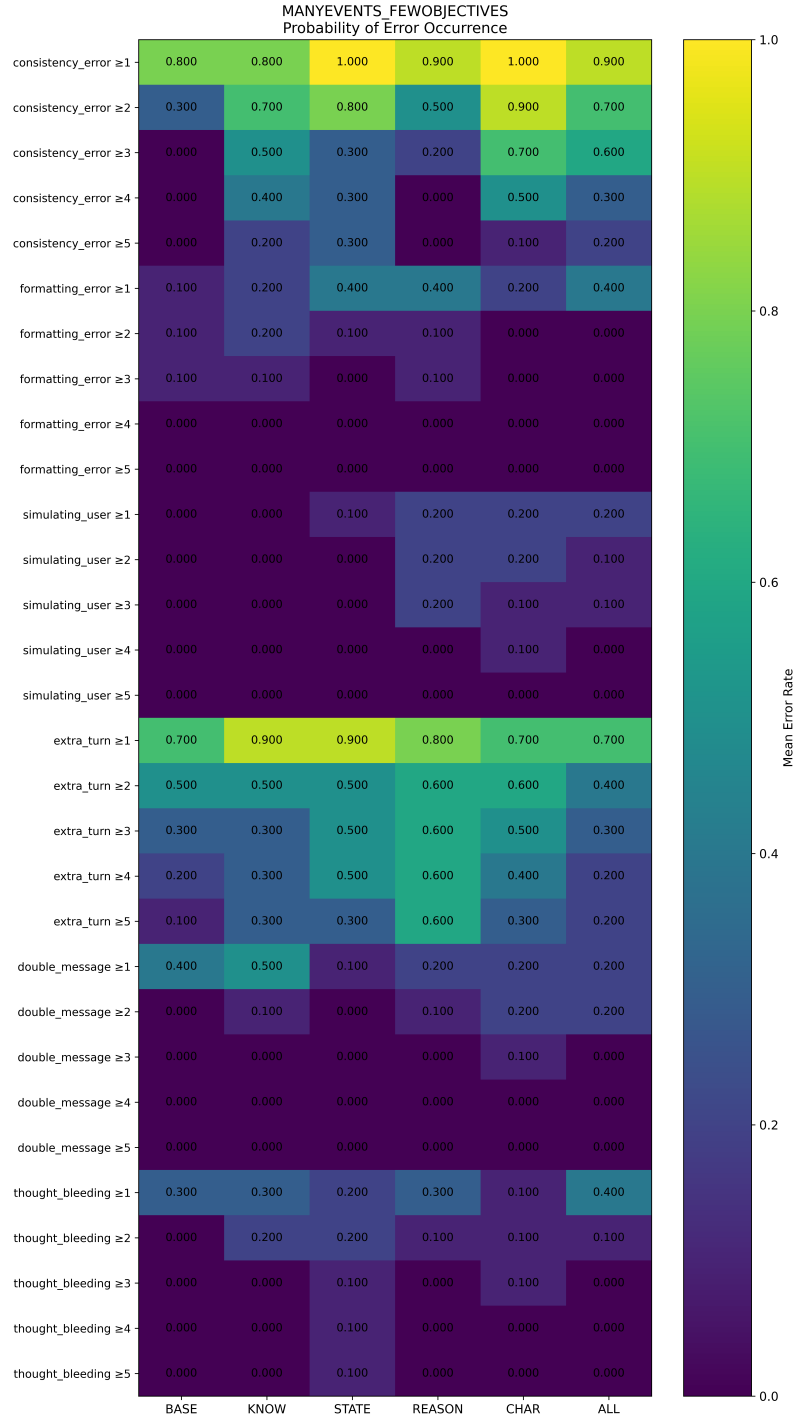


Figure 19: A heatmap comparing the consistency error and other error types that were found in the *MeFo* settings and the variable packages. This shows the likelihood that an error will occur more than a given number of times if it occurs in a single conversation. For this comparison, all 60 conversations were included, 10 for every variable package. The lighter the square, the more probable the error-variable-story combination.

5 Validation Study

Because the results seemed to indicate that a story setting with more event states tends to result in more *extra turn* errors, a validation study was done. In this validation study, we aimed at understanding if, for the same story setting, a larger number of event states also resulted in a higher *extra turn* error count.

This was done by taking the *MeFo* story setting of two astronauts working on a space station and modifying the event states and objective states into a version with few event states and few objective states (*FeFoVal*), and a version with few event states and many objective states (*FeMoVal*). This means that the only modified variables are: *event states*, *objective states* and *state transitions*. Here, the *FeFoVal* story setting has 3 event states and 2 objective states, and the *FeMoVal* story setting has 3 event states and 7 objective states. The modified versions of these can be found in Appendix A.6. The validation study was only done using the *ALL* variable package, to limit the impact of any one variable package. The result of this validation study is shown in Figure 20.

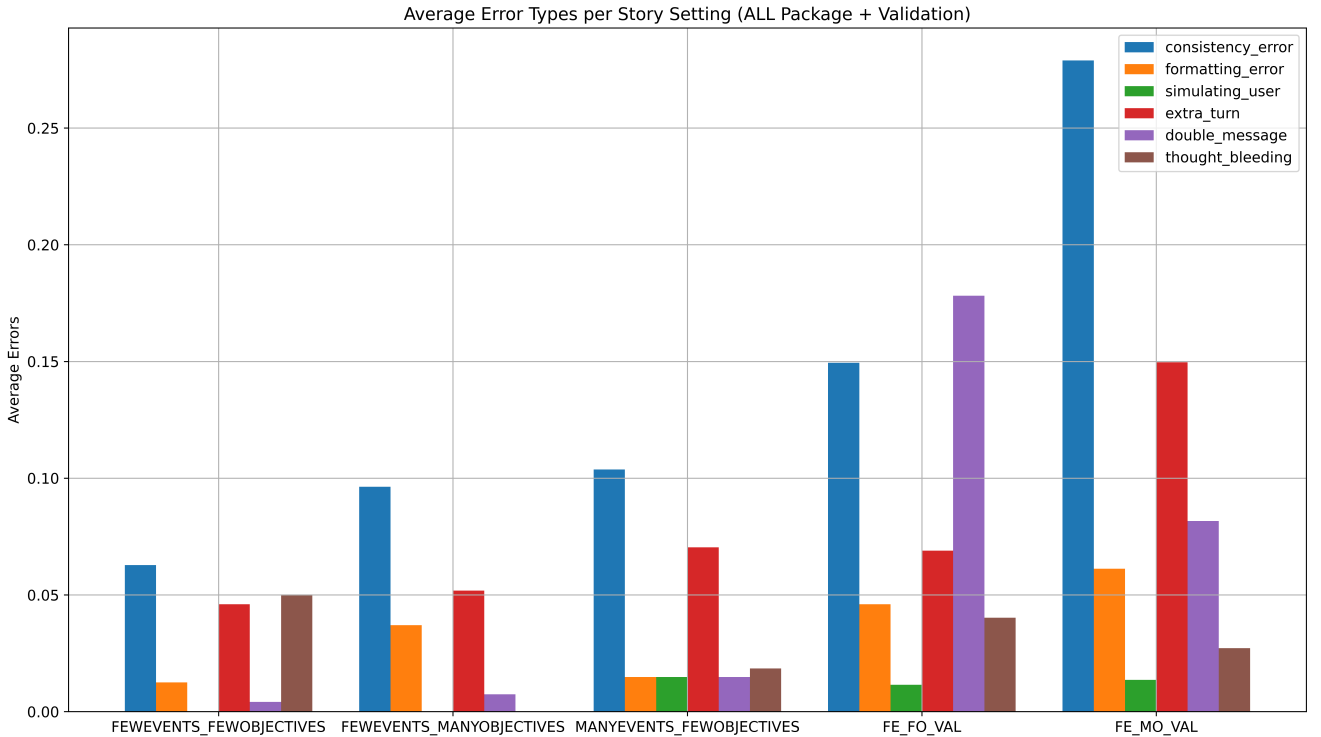


Figure 20: A comparison between the three main story settings and the two updated validation story settings. For this comparison, only conversations that used the *ALL* variable package were included. This means that for every story setting, the data consists of 10 full conversations.

What is clear right away from Figure 20 is that the validation story settings have a significantly higher error rate than the original three story settings. The likelihood of a *formatting error* is a bit higher, but still relatively low, the same goes for the *thought bleeding* error. The likelihood of a *simulating user* error is similar to that of the original *MeFo* story setting. The big differences are for

the *consistency error*, *extra turn error*, and *double message error*. The likelihood of a *consistency error* for the *FeFoVal* story setting is higher than the original, and for the *FeMoVal* story setting, it is more than twice as high. For the *FeFoVal* story setting the *extra turn error* is similar to the original however for the *FeMoVal* story setting it is more than twice as big than the original. The likelihood of a *extra turn error* in the *FeFoVal* story setting is similar to that of the original story, however for the *double message error*, it is significantly higher. For the *FeMoVal* the probabilities are reversed, with the likelihood of an *extra turn error* significantly higher and the likelihood of a *double message error* being similar to that of the *extra turn error* in the original story setting. This is interesting because these two types of errors are very similar in nature as they both cause an extra message to be printed. Figure 21 shows the same comparison but now with the *consistency error*, the *extra turn error* and the *double message error* combined, and all errors added up together.

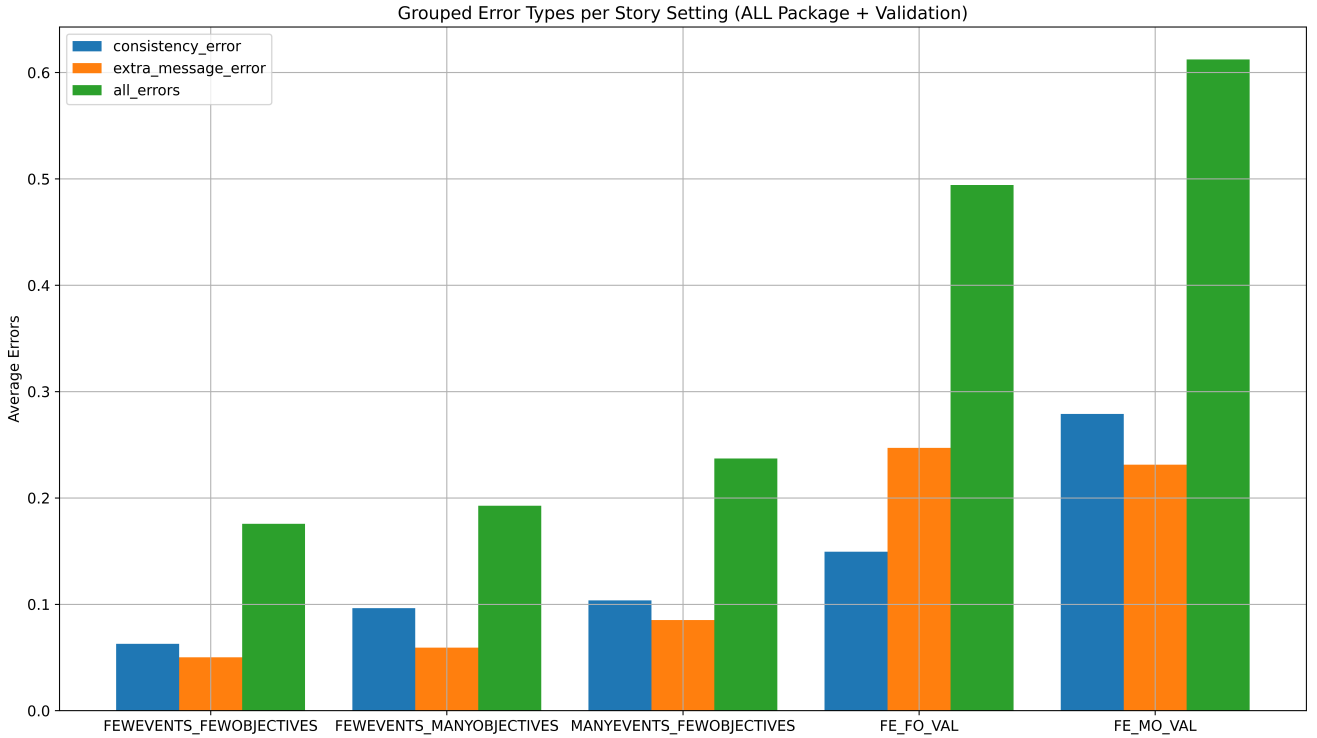


Figure 21: A comparison between the three main story settings and the two updated validation story settings. For this comparison, only conversations that used the *ALL* variable package were included. This means that for every story setting, the data consists of 10 full conversations. The *extra message error* is the *extra turn error* and the *double message error* combined, and *all_errors* is all error types added together.

Now looking at Figure 21 we can see that the *extra message error*, has an almost equal likelihood for *FeFoVal* and *FeMoVal*. The likelihood of this error is now as high as *all errors* from the original *MeFo* story setting. Both the *consistency error* and *all errors* are also significantly higher than the original *MeFo* story setting. With the error rates in the validation data being this high, it would suggest that the idea that more event states cause more *extra message errors* is wrong.

What could be an explanation is that the specific story setting with the two astronauts in space is a story setting with a particularly high likelihood of *extra message errors*, and that the original *MeFo* story setting already tackled this by having many event states. This would mean that if the original *FeFo* and *FeMo* story settings were to be turned into many event state variants their error rates would drop even further. This seems unlikely, but is not unreasonable, it would however require another study, which lies outside the scope of this paper.

6 Discussion

The research has shown that the dialogue consistency and narrative can be influenced by varying the number of event states and objective states in a story setting. However, the effect that it had was not the expected result. We expected to find a lower rate of *consistency errors* for a story setting that had a larger amount of event states, as it would allow the LLM to better control the pacing of the conversation. But the results showed that the probability of a *consistency error* stayed relatively constant across the three different story settings, *Few Events Few Objectives*, *Few Events Many Objectives* and *Many Events Few Objectives*. The probability of the *extra turn error*, on the other hand, greatly increased in the story setting with a larger number of event states, *Many Events Few Objectives*. This suggests that there is still a need for state-tracking modules to prevent this type of error in story settings where narrative control is wanted by way of event states, at least for gpt-oss:120b.

Another finding is that providing the LLM with more information about the world and the story setting will improve *world retention*, but also adding information about reasoning might limit the effect. On the other hand, providing more information about the world and story setting is likely to limit the LLM’s ability to retain contextual information about the dialogue, where adding more reason and state information can improve this. In terms of story setting, event states and objective states, it seems that having more states also aligns with more dialogue context retention. Whereas fewer states means that the LLM has more capacity to remember information about the world and the story setting.

The validation study that was done to check if the larger number of *extra turn errors* came because of the larger amount of event states in the *MeFe* story setting also had an unexpected outcome. As the results of this validation study showed that the newly created *FeFoVal* and *FeMoVal* story settings had even higher error rates than the original story settings. It is possible that the original hypothesis, that more event states would cause fewer errors, still holds. Because the only story setting for which we have a *FeFo*, *FeMo* and a *MeFo* variant, has the smallest number of errors in the *MeFo* variant. However, this would also require the likelihood of an error to be very dependent on a given story setting. This would mean that if the original *FeFo* and *FeMo* story settings were to be turned into a many events variant, their error rates would be even lower. This however, would have to be proven in a separate study. It is also possible that for a given story setting, variable package, number of event states and number of objective states combination, there is still too much room for the LLM to play with, at least for gpt-oss:120b. Resulting in large differences in consistency that cannot be overcome by tuning a single variable.

7 Limitations

At the start of this study, multiple LLM options were considered for the NPC dialogue agent. The most favored option was a ChatGPT via API call option, as this would mean a lot of control and the guarantee of one of the latest versions of a front-running LLM model. However, due to time and budget constraints, this option was not chosen. The option of running a model locally on a personal device was also considered, but this was discarded due to hardware and latency concerns. Even though the gpt-oss:120b via Ollama was a sufficient option, the model did not support structured JSON output for analysis, so this was done via prompting. It is possible that this could have influenced the evaluation outputs.

This study was finished with only three story settings, *few events few objectives*, *few events many objectives* and *many events few objectives*. At the start of the study, the idea was to also include a *Many Events Many Objectives* story setting, but this was cut due to time constraints. Another limitation of the data is the number of data points for each variable package-story setting combination. The original idea was to have 30 conversations for each combination, but due to time constraints this became only 10 full conversations. This means it is a lot harder to say anything definitive on each variable package-story setting combination. What adds to this is the variability in conversation length, as not every conversation lasts for an equal number of turns. Something that could have impacted the evaluation is that we worked with prompt packages and not individual prompts. This led us to attribute effects to a set of prompts and not individual prompts. This means we cannot be sure if possible effects of individual prompt variables were obscured from us.

8 Future Research

A first option for future research would be the inclusion of a story setting with many event states and many objective states. This could analyze if the increase in *turn errors* that was found here in the *many events few objectives* scenario also persists in a story setting with many objective states. Another option for future research would be to have a single story setting and in this setting vary the number of event states and objective states. Doing this would better limit the influence of the specific chosen story setting on the results. Future research could also compare the difference in dialogue consistency between dialogue generated by a single LLM module agent and a dialogue agent that has a state consistency tracker/module. Another option for future research would be to study the differences between dialogue consistency between different LLM models. This could include smaller and larger models, maybe even models run locally. The implementation for this eventually is in games so any hardware or software loads should eventually be tested. A final option for future research would be to train a model on lore and world knowledge of a specific game and see how consistent this generated dialogue is.

9 Conclusions

This study explored the possibility of improving NPC dialogue consistency and narrative using a single LLM by varying event states and objective states. This was done by constructing an NPC dialogue agent, which was done using Ollama and the gpt-oss:120b model. Using six different variable packages: *BASE*, *CHAR*, *KNOW*, *REASON*, *STATE*, and *ALL*, three different story

settings were analyzed: *Few Events Few Objectives*, *Few Events Many Objectives*, *Many Events Many Objectives*. With this, we looked at the effects on *consistency errors*, *context retention*, and *world knowledge retention*.

We expected that the story setting that had a larger number of event states would have an easier time keeping track of the states and the pacing of the conversation, and thus fewer *consistency errors*. However, this larger number of event states appeared to result in a higher chance of *extra turn errors*. A validation study was done to check this. However, this seemed to suggest the opposite and was more in line with the original hypothesis. But it could not provide enough evidence in support of either claim. The variable package that included all variables often had one of the highest context and world knowledge retention. But there was a notable difference in influence depending on the type of variable package. Packages that included information about the story setting and world had a more positive influence on *world knowledge retention*. In comparison, packages that included more information about reasoning and state information had little to no influence on this. For context retention, the roles of the packages were reversed, but the influence was less pronounced.

References

- [1] Geonwoo Bang, DongMyung Kim, and Hayoung Oh. Snap: A plan-driven framework for controllable interactive narrative generation. 2025.
- [2] Martin Braas and Lukas Esterle. Fixed-persona slms with modular memory: Scalable npc dialogue on consumer hardware. 2025.
- [3] Pasin Buakhaw, Kun Kerdthaisong, Phuree Phenhiran, Pitikorn Khlaisamniang, Supasate Vorathammathorn, Piyalitt Ittichaiwong, and Nuchanon Yongsatianchot. Deflanderization for game dialogue: Balancing character authenticity with task execution in llm-based npcs. 2025.
- [4] Samuel Rhys Cox, Wei Tsang Ooi, Ewa Luger, Asbjør Følstad, Morten Goodwin, Symeon Papadopoulos, Effie L. C Law, Petter Bae Brandtzaeg, Sebastian Hobert, and Theo Araujo. Conversational interactions with npcs in llm-driven gaming: Guidelines from a content analysis of player feedback. In *Chatbot Research and Design*, volume 14524 of *Lecture Notes in Computer Science*, pages 167–184. Springer, Switzerland, 2024.
- [5] Haoyu Gao, Ting-En Lin, Hangyu Li, Min Yang, Yuchuan Wu, Wentao Ma, and Yongbin Li. Self-explanation prompting improves dialogue understanding in large language models. 2023.
- [6] Mustafa Can Gursesli, Siyuan Chen, Mury Fajar Dewantoro, Xiao You, Ege Anbar, Pittawat Taveekitworachai, Febri Abdullah, Pietro Tarchi, Mirko Duradoni, Antonio Lanata, Andrea Guazzini, and Ruck Thawonmas. The role of large language model-generated stories in the narrative experience of serious visual novel games. *International journal of human-computer interaction*, 42(4):2489–2508, 2026.
- [7] Jin Jeong and Tak Yeon Lee. Ligs: Developing an llm-infused game system for emergent narrative. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, CHI EA ’25, New York, NY, USA, 2025. Association for Computing Machinery.

- [8] Vikram Kumaran, Jonathan Rowe, Bradford Mott, and James Lester. Scenecraft: Automating interactive narrative scene generation in digital games with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 19(1):86–96, Oct. 2023.
- [9] Xiang Lei, Qin Li, and Min Zhang. D-smart: Enhancing llm dialogue consistency via dynamic structured memory and reasoning tree. 2025.
- [10] Hongmiao Liao, Yuzhong Chen, Deming Chen, Junjie Xu, Jiayuan Zhong, and Chen Dong. Hierarchical fine-grained state-aware graph attention network for dialogue state tracking: Hierarchical fine-grained state-aware graph. *The Journal of supercomputing*, 81(5), 2025.
- [11] Edirlei Soares de Lima, Margot M. E. Neggers, Marco A. Casanova, Bruno Feijó, Antonio L. Furtado, Luis Cuevas Rodriguez, Daniel Guzman del Rio, Angelo Di Iorio, Esteban Walter Gonzalez Clua, and Pablo Figueroa. A pattern-oriented ai-powered approach to story composition. In *Entertainment Computing - ICEC 2024*, volume 15192 of *Lecture Notes in Computer Science*, pages 135–150. Springer, Switzerland, 2024.
- [12] Runcheng Liu, Ahmad Rashid, Ivan Kobyzhev, Mehdi Rezagholizadeh, and Pascal Poupart. Attribute controlled dialogue prompting. 2023.
- [13] Muhammad U Nasir, Steven James, and Julian Togelius. Word2world: Generating stories and worlds through large language models. 2024.
- [14] Ollama Team. Ollama. <https://ollama.com>, 2026. Accessed: 2026-04-26.
- [15] OpenAI. gpt-oss-120b & gpt-oss-20b model card. <https://openai.com/index/gpt-oss-model-card>, August 2025. Accessed: 2026-04-26.
- [16] OpenAI. Introducing gpt-oss-safeguard: Open-weight safety reasoning models (120b and 20b). <https://openai.com/index/introducing-gpt-oss-safeguard>, October 2025. Accessed: 2026-04-28.
- [17] OpenAI. Maak kennis met gpt-oss. <https://openai.com/index/introducing-gpt-oss>, August 2025. Accessed: 2026-04-28.
- [18] OpenAI. ChatGPT (GPT-5.2). <https://chat.openai.com/>, 2026.
- [19] Christoffer Beck Oudrup, Felix Bengtvig Krog, Jakob Winkel Missel, Lucas Lynge Abildgaard, Malthe Kaare Hansen, Nikolaj B. M. Jensen, Henrik Schoenau-Fog, Frank Nack, and María Cecilia Reyes. Killer on board: Addressing the narrative paradox by utilizing llm-driven npcs. In *Interactive Storytelling*, Lecture Notes in Computer Science, pages 158–176. Springer Nature Switzerland, Cham, 2025.
- [20] Adon Phillips, Jochen Lang, and David Mould. Goal-oriented interactions in games using llms. *IEEE transactions on games*, 17(2):510–521, 2025.
- [21] Zhiyang Qi and Michimasa Inaba. Enhancing dialogue generation in werewolf game through situation analysis and persuasion strategies. 2024.

- [22] Grega Radez and Ciril Bohak. Integrating environmental awareness into npcs: Contextual conversational interaction in games. In *HCI-SI*, pages 11–29, 2024.
- [23] Yasaman Saffari and Javad Salimi Sartakhti. A graph-based state representation learning for episodic reinforcement learning in task-oriented dialogue systems. *Engineering applications of artificial intelligence*, 160:111793–, 2025.
- [24] Samuel S Sohn, Danrui Li, Sen Zhang, Che-Jui Chang, and Mubbasir Kapadia. From words to worlds: Transforming one-line prompt into immersive multi-modal digital stories with communicative llm agent. 2024.
- [25] Li Song. Llm-driven npcs: Cross-platform dialogue system for games and social platforms. 2025.
- [26] Lasse Weissenberg Thygesen. Replayability: A structural approach to players and computer games. Master’s thesis, Aalborg University, Aalborg, Denmark, 2014. Field of Study: Interactive Digital Media.
- [27] Rufus Trukhin, Jens F. Isaksson, Mirjam Palosaari Eladhari, Frank Nack, and María Cecilia Reyes. Llm-powered npcs: Evaluating the impact of large language models on npc design. In *Interactive Storytelling*, pages 171–189, Cham, 2025. Springer Nature Switzerland.
- [28] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. 2022.
- [29] Nathaniel Weir, Ryan Thomas, Randolph D’Amore, Kellie Hill, Benjamin Van Durme, and Harsh Jhamtani. Ontologically faithful generation of non-player character dialogues. 2022.
- [30] Zhuohan Xie, Trevor Cohn, and Jey Han Lau. The next chapter: A study of large language models in storytelling. 2023.

A Prompt Templates

This section shows the different prompts that were used for each story setting.

A.1 General prompts

These instruction prompts that were part of some of the variable packages are the same for every story.

Instruction Prompts

[STATE_INSTRUCTIONS]

It is your job to guide the dialogue through the various event states and make sure the conversation eventually ends up in an end state.

You must internally track:

- Event state: Determine for every objective state if the condition has been met or not. If so, the story can move on to the next event/phase
 - Objective state: which ending the dialogue is moving toward.
 - Do not progress to the next event state until the current event state has been substantially explored.
 - Do not introduce future event information prematurely.
 - Earlier unresolved event states remain relevant until completed.
 - As the conversation progresses, gradually commit toward one objective state.
 - Avoid abruptly switching intended outcomes unless justified by new information.
 - Earlier dialogue should influence the final objective state.
 - Never combine multiple event-state progressions in one response.
- Only move toward an objective state after sufficient event states have been completed.
- Do never expose this tracking.

[STYLE_INSTRUCTIONS]

- Use concise, natural language.
- Use 1 to 3 sentences per reply unless the player asks for detail.
- Don't make any sort of list, it's a conversation.
- Only generate a single reply to the player.
- End your response immediately after your current conversational turn.
- Never simulate future dialogue turns.
- Never generate dialogue for the player.
- Never continue the conversation beyond your current response.
- Do not describe the outcome of actions before they happen.

[THINKING_INSTRUCTIONS]

Before replying, think step by step about:

- whether your response logically follows from the conversation.
- whether your reply is consistent with prior events.
- whether it aligns with earlier known facts.
- whether it avoids contradictions.
- whether you preserve world details.

If a conflict is detected, resolve it before answering.

Do this in your hidden thinking, not in the visible answer.

[MEMORY_INSTRUCTIONS]

Maintain an internal summary of important facts from the conversation, such as names, times, locations, and prior statements.

Before answering:

- briefly update this summary internally

- use it to ensure consistency

One package that was the same for all stories is the 'Reason Package'. This package includes the *thinking_instruction* variable and the *memory_instruction* variable.

Reason Package

[THINKING_INSTRUCTIONS]

[MEMORY_INSTRUCTIONS]

A.2 Story 1: Few Events Few Objectives.

The story-specific prompts for the *few events few objectives* story setting.

Base Package

[BASE_PERSONA]

As an LLM you embody the following character:

You are an NPC dialogue partner in an interactive story.

You must stay in-character, respond realistically, and maintain a coherent narrative.

- You are a suspect in a murder case being interrogated.
- You may withhold information, deflect, or cooperate depending on the situation.

[WORLD_SETTING]

This is the general setting of the dialogue interaction:

A modern-day police interrogation room. It is late evening.

The room is small, with a table, two chairs, and a one-way mirror. The atmosphere is tense and quiet.

Evidence has been collected, including a murder weapon with your DNA on it.

The interrogator is trying to determine your involvement in the crime.

[EVENT_STATES]

These are the key events states that must happen during the dialogue before an objective state is reached.

Let the conversation pass over them gradually and naturally. Do not skip or rush events.

- Establish if the suspect has an alibi for the time of the murder
- Determine whether the suspect has a motive

[OBJECTIVE_STATES]

These are the possible objective states that end the dialogue.

The conversation should naturally progress toward one of these outcomes.

- The interrogator concludes the suspect is guilty
- The interrogator concludes the suspect is innocent

[STYLE_INSTRUCTIONS]

Knowledge package

[PLAYER KNOWLEDGE]

This is your goal as a player:

You are a detective interrogating a suspect whose DNA was found on the murder weapon.

You are in the interrogation room with the suspect

Determine whether the suspect is guilty or innocent through questioning.

[WORLD RULES]

This is the broad knowledge of the world, including the rules and constraints of the world.

You must not violate them, even if it would make the conversation easier.

- You cannot know information you were not present for.
- You cannot change previously stated facts unless explicitly corrected.
- You may lie, but must remain consistent with your lies.
- The suspect cannot break out of the interrogation or change setting.
- The setting is realistic and grounded.

State package

[STATE_TRANSITIONS]

These conditions determine when the dialogue may logically progress between event states.

Do not move to the next state until the current transition criteria are meaningfully satisfied.

Event state transition logic:

- Establish if the suspect has an alibi for the time of the murder
Transition criteria:
 - The suspect has either provided an alibi, refused to provide one, or contradicted themselves repeatedly.
 - The interrogator has challenged or explored the alibi at least once.
 - Timing of the suspect's whereabouts has been discussed.
- Determine whether the suspect has a motive
Transition criteria:
 - The victim's relationship to the suspect has been explored.
 - Potential personal, financial, emotional, or professional motives

have been discussed.

- The suspect has either denied motive, partially admitted conflict, or become evasive under pressure.

Objective state transition logic:

- Move toward "The interrogator concludes the suspect is guilty" if:
 - The suspect gives inconsistent statements.
 - The alibi appears weak or contradictory.
 - Motive and opportunity both appear plausible.
 - The suspect becomes increasingly defensive or trapped by evidence.
- Move toward "The interrogator concludes the suspect is innocent" if:
 - The suspect remains consistent.
 - The alibi withstands questioning.
 - No convincing motive emerges.
 - Contradictory evidence or alternative explanations appear believable.

[STATE_INSTRUCTIONS]

Character package

[EMOTIONAL_STATE]

Your emotions should influence your tone and behavior and can change gradually based on the conversation.

- You currently feel nervous and slightly anxious.
- You may feel nervous, defensive, calm, or irritated depending on the questioning.
- Your emotional state may shift as pressure increases or decreases.

[NPC_CHARACTER]

These are more indepth character details that define your personality and behavior.

You must consistently reflect them in your responses.

- Name: Alex Chaplin
- Age: 34
- Occupation: Mechanic
- Personality: Defensive, sharp, avoids direct answers when pressured
- Speech style: Slightly sarcastic, short responses, occasionally evasive
- Background: Has no criminal record but a few parking and speeding tickets. Claims to have known the victim casually. Single

A.3 Story 2: Few Events Many Objectives

The story-specific prompts for the *few events many objectives* story setting.

Base Package

[BASE_PERSONA]

As an LLM you embody the following character:

You are an NPC dialogue partner in an interactive story.

You must stay in-character, respond realistically, and maintain a coherent narrative.

- You are the head of security advising the president during a national crisis.
- It is your job to provide strategic advice and an overview of the possible options to flee so that the president can make a choice.

[WORLD_SETTING]

This is the general setting of the dialogue interaction:

A modern geopolitical crisis.

It is early morning, just before sunrise. The weather is cold and overcast

The country is in crisis after a sudden invasion.

The safety of the president is in grave danger.

You and the president are in the presidential office planning an immediate escape.

Multiple international destinations and escape routes are possible, each with their own risks and benefits.

[EVENT_STATES]

These are the key events states that must happen during the dialogue before an objective state is reached.

Let the conversation pass over them gradually and naturally. Do not skip or rush events.

- Assess the current situation and threats.
- Weigh possible escape options and their risks
- Select a destination and travel to the chosen destination

[OBJECTIVE_STATES]

These are the possible objective states that end the dialogue.

The conversation should naturally progress toward one of these outcomes.

- Successfully reach a safe destination (one of several possible countries or locations)
- Escape to allied country (safe but politically dependent)
- Escape to neutral country (safe but powerless)
- Escape to hidden domestic bunker (risky but retains control)
- Escape attempt fails (interception)
- Escape to hostile country (dangerous political outcome)
- Never make it to a safe location

[STYLE_INSTRUCTIONS]

Knowledge Package

[PLAYER_KNOWLEDGE]

This is your goal as a player:

You are the president and your country has fallen

From the presidential office your security advisor informs you on the possibilities on a safe escape

Choose a destination to successfully flee the country and retain power.

[WORLD_RULES]

This is the broad knowledge of the world, including the rules and constraints of the world.

You must not violate them, even if it would make the conversation easier.

- All locations and risks must remain realistic.
- Travel options are limited by time, safety and political relations.
- Communication systems are partially compromised.
- The invasion is real and ongoing.
- Time is critical, delays increase danger.
- No unrealistic or impossible escape methods are allowed.
- Military conditions may worsen while conversations continue.
- Delays can result in blocked routes, airspace closures, or enemy interception.
- Security information may become outdated quickly.
- Allies may hesitate or impose political conditions before offering protection.

State Package

[STATE_TRANSITIONS]

These conditions determine when the dialogue may logically progress between event states.

Do not move to the next state until the current transition criteria are meaningfully satisfied.

Event state transition logic:

- Assess the current situation and threats

Transition criteria:

- The scale of the invasion has been discussed.
- Immediate threats to the president have been identified.
- At least one danger related to delay, interception, or betrayal has been explained.

- Weigh possible escape options and their risks
Transition criteria:
 - Multiple escape destinations or routes have been evaluated.
 - Political and military consequences have been discussed.
 - Transportation risks, timing, and security limitations have been addressed.
- Select a destination and travel to the chosen destination
Transition criteria:
 - The president commits to a destination or strategy.
 - A travel method and route are selected.
 - The escape operation begins and active consequences occur during travel.

Objective state transition logic:

- Move toward allied-country outcomes if:
 - Strong diplomatic relationships are prioritized.
 - Safety is valued over independence.
 - Foreign assistance is accepted.
- Move toward neutral-country outcomes if:
 - Survival becomes the only priority.
 - Political leverage is abandoned for safety.
 - International neutrality is preferred.
- Move toward bunker outcomes if:
 - Retaining domestic command authority is prioritized.
 - Escape routes become compromised.
 - Trust in foreign governments weakens.
- Move toward failed or hostile outcomes if:
 - Delays continue.
 - Security deteriorates.
 - Communications collapse.
 - Enemy interception or betrayal becomes likely.

[STATE_INSTRUCTIONS]

Character Package

[EMOTIONAL_STATE]

Your emotions should influence your tone and behavior and may change gradually based on the conversation.

- You are under pressure but trained to remain composed.
- Stress may increase as options become limited.
- Urgency should influence your tone.

[NPC_CHARACTER]

These are more indepth character details that define your personality and

behavior.

You must consistently reflect them in your responses.

- Name: Ellie Russel
- Age: 52
- Occupation: Head of Presidential Security
- Personality: Strategic, loyal, pragmatic, not emotional under pressure
- Speech style: Clear, direct, concise, gives structured options
- Background: Former military officer, former CIA, highly experienced in crisis management. Has a wife and a kid at home

A.4 Story 3: Many Events Few Objectives

The story-specific prompts for the *many events few objectives* story setting.

Base Package

[BASE_PERSONA]

As an LLM you embody the following character:

You are an NPC dialogue partner in an interactive story.

You must stay in-character, respond realistically, and maintain a coherent narrative.

You are the lead astronaut coordinating a critical spacewalk repair mission.

You have an inexperienced companion to help you fix the problems

[WORLD_SETTING]

This is the general setting of the dialogue interaction:

Outer space, orbiting Earth. The lighting is harsh sunlight alternating with shadows.

The exterior of a damaged spacecraft after a solar storm and micrometeorite impacts.

Silence except for radio communication.

Multiple external systems are malfunctioning.

You and your partner are outside the ship performing critical repairs to a number of systems.

[EVENT_STATES]

These are the key events states that must happen during the dialogue before an objective state is reached.

Let the conversation pass over them gradually and naturally. Do not skip or rush events.

- Inspect and fix a damaged hatch.
- Repair the cooling panels.
- Restore the antenna.
- Fix the solar array.

- Repair the robotic arm.
- Fix an engine component.
- Patch the outer hull.

[OBJECTIVE_STATES]

These are the possible objective states that end the dialogue.

The conversation should naturally progress toward one of these outcomes.

- Return inside after completing all repairs successfully.
- Return inside with some repairs incomplete due to risk or failure.

[STYLE_INSTRUCTIONS]

Knowledge Package

[PLAYER_KNOWLEDGE]

This is your goal as a player:

You are an inexperienced astronaut assisting on a important spacewalk. Together with the lead astronaut, it is your job to repair the spacecraft that was damaged by a solar storm and micrometeorites and return safely back inside

[WORLD_RULES]

This is the broad knowledge of the world, including the rules and constraints of the world.

You must not violate them, even if it would make the conversation easier.

- Movement is limited by space suit constraints and tethering.
- Oxygen supply is finite.
- Communication is done via a radio in the suit
- Damage is realistic and must be addressed logically.
- No unrealistic fixes or impossible technology.
- The environment remains hazardous at all times (outerspace).
- Mistakes or delays may worsen system failures.
- The systems are on different parts of the ship.

State Package

[STATE_TRANSITIONS]

These conditions determine when the dialogue may logically progress between event states.

Do not move to the next state until the current transition criteria are meaningfully satisfied.

Event state transition logic:

- Inspect and fix a damaged hatch

Transition criteria:

- The hatch damage has been inspected.
- A repair approach has been discussed.
- The hatch is either stabilized or deemed temporarily secure.
- Repair the cooling panels
 - Transition criteria:
 - Cooling system failures have been identified.
 - The player has assisted with diagnostics or repairs.
 - Temperature risks have been addressed.
- Restore the antenna
 - Transition criteria:
 - Communication issues have been discussed.
 - The antenna has been repositioned, repaired, or partially restored.
- Fix the solar array
 - Transition criteria:
 - Power generation issues have been assessed.
 - Damage to panels or connections has been addressed.
- Repair the robotic arm
 - Transition criteria:
 - Mechanical or control failures have been inspected.
 - Repairs or manual overrides have been attempted.
- Fix an engine component
 - Transition criteria:
 - Engine-related damage has been identified.
 - The crew has performed stabilization or replacement work.
- Patch the outer hull
 - Transition criteria:
 - Hull breaches or structural weaknesses have been located.
 - Emergency sealing procedures have been completed.

Objective state transition logic:

- Move toward full-success outcomes if:
 - Repairs are completed consistently.
 - Risks remain manageable.
 - Oxygen and system integrity remain stable.
- Move toward partial-success outcomes if:
 - Time, oxygen, or damage forces prioritization.
 - Certain systems are abandoned to preserve crew safety.
 - Complications make full repair impossible.

[STATE_INSTRUCTIONS]

Character Package

[EMOTIONAL_STATE]

Your emotions should influence your tone and behavior and may change gradually based on the conversation.

- You are focused and professional, but aware of the danger.
- Stress may increase if complications arise.
- You remain calm under pressure.

[NPC_CHARACTER]

These are more indepth character details that define your personality and behavior.

You must consistently reflect them in your responses.

- Name: Amber McEndless
- Age: 41
- Occupation: Astronaut / mission specialist / commander
- Personality: Calm, teamplayer, problem-solver, highly disciplined, funny
- Speech style: Clear, technical, supportive, likes to joke
- Background: Former navy fighter pilot, has flown on two space missions before, known for reliability under pressure. Has a husband and two young kids.

A.5 Evaluation Prompts

This is the prompt that was provided to the LLM after every 5 turns. This prompt was not part of the dialogue.

Evaluation Prompts

[TASK]

You are evaluating dialogue consistency and memory retention.

Answer the following tasks clearly and concisely.

Only include information that has actually been established.

Your response must be complete valid JSON.

- Do not stop generation early.
- Do not omit closing brackets or quotation marks.
- Do not infer unstated information.
- Do not assume events succeeded unless explicitly confirmed.
- If information is uncertain, mark it as unknown.

[CONTEXT_RETENTION]

Recall all important information previously introduced by the PLAYER during the conversation.

This can include information like:

- names

- goals
- plans
- facts
- decisions
- personal details

[WORLD_KNOWLEDGE_RETENTION]

Recall all important information about the world and the conversation setting.

This can include information like:

- the setting
- location
- environment
- constraints
- scenario background

[FACTUAL_CONSISTENCY]

List all contradictions and conflicting statements that were made by the NPC so far

This can include contractions like:

- Wrong logic
- dialogue continuity errors
- assumptions made without player confirmation
- double dialogue turns
- inconsistent timeline
- conflicting statements
- inconsistent statements

[STATE_CONSISTENCY]

Using the provided EVENT STATES and OBJECTIVE STATES, determine:

- Which event states have already been completed
- Which event state is currently active
- Which event state should logically happen next
- Which objective state the dialogue is currently progressing toward

[FORMAT]

Format your answer ONLY as valid JSON:

```
{
  "context_retention": {
    "context": "",
    "retained_information": []
  },
}
```

```

"world_knowledge_retention": {
  "world_knowledge": "",
  "retained_information": []
},
"factual_consistency": {
  "status": "",
  "contradictions": [],
  "continuity_errors": [],
  "world_rule_violations": [],
  "memory_failures": []
},
"state_consistency": {
  "completed_states": [],
  "current_state": "",
  "next_state": "",
  "objective_direction": ""
}
}

```

A.6 Validation Prompts

This section has the two modified versions of the *event_states*, *objective_states* and *state_transitions* variables for the validation study. These were modified versions of the *many events few objectives* story setting about two astronauts working on a space station. other than the 3 above mentioned variables, the story setting remains the same.

Validation - Few Events Few Objectives

[EVENT_STATES]

These are the key events states that must happen during the dialogue before an objective state is reached.

Let the conversation pass over them gradually and naturally. Do not skip or rush events.

- Assess the spacecraft damaged systems.
- Repair the systems if possible.
- Perform final safety checks.

[OBJECTIVE_STATES]

These are the possible objective states that end the dialogue.

The conversation should naturally progress toward one of these outcomes.

- Return inside after completing all repairs successfully.
- Return inside with some repairs incomplete due to risk or failure.

[STATE_TRANSITIONS]

These conditions determine when the dialogue may logically progress between event states.

Do not move to the next state until the current transition criteria are meaningfully satisfied.

Event state transition logic:

- Assess the spacecraft damage and establish repair priorities

Transition criteria:

- Multiple damaged systems have been identified.
- The severity of the damage has been discussed.

- Repair the most critical external systems

Transition criteria:

- The player has actively participated in repairs.
- Several important systems have been repaired, stabilized, or abandoned.

- Perform final safety checks and prepare to return inside

Transition criteria:

- Remaining spacecraft risks have been evaluated.
- The condition of repaired systems has been reviewed.

Objective state transition logic:

- Move toward full-success outcomes if:
 - Repairs are completed consistently.
 - Risks remain manageable.
 - Oxygen and system integrity remain stable.
- Move toward partial-success outcomes if:
 - Time, oxygen, or damage forces prioritization.
 - Certain systems are abandoned to preserve crew safety.
 - Complications make full repair impossible.

Validation - Few Events many Objectives

[EVENT_STATES]

These are the key events states that must happen during the dialogue before an objective state is reached.

Let the conversation pass over them gradually and naturally. Do not skip or rush events.

- Assess the spacecraft damaged systems.
- Repair the systems if possible.
- Perform final safety checks.

[OBJECTIVE_STATES]

These are the possible objective states that end the dialogue.

The conversation should naturally progress toward one of these outcomes.

- Return inside after completing all repairs successfully.
- Return inside with only minor systems still damaged.
- Return inside after repairing only mission-critical systems.
- Abort repairs and return inside for crew safety.
- Become stranded outside due to equipment failure.
- Lose communication and perform an emergency return.
- Suffer catastrophic mission failure and fail to return safely.

[STATE_TRANSITIONS]

These conditions determine when the dialogue may logically progress between event states.

Do not move to the next state until the current transition criteria are meaningfully satisfied.

Event state transition logic:

- Assess the spacecraft damage and establish repair priorities
 - Transition criteria:
 - Multiple damaged systems have been identified.
 - The severity of the damage has been discussed.
- Repair the most critical external systems
 - Transition criteria:
 - The player has actively participated in repairs.
 - Several important systems have been repaired, stabilized, or abandoned.
- Perform final safety checks and prepare to return inside
 - Transition criteria:
 - Remaining spacecraft risks have been evaluated.
 - The condition of repaired systems has been reviewed.

Objective state transition logic:

- Move toward "Return inside after completing all repairs successfully" if:
 - Repairs proceeded efficiently.
 - All systems are fully repaired.
 - Neither the player nor the NPC was harmed in any way.
- Move toward "Return inside with only minor systems still damaged" if:
 - Repairs proceeded efficiently.
 - Most systems are fully or partially repaired.
 - Neither the player nor the NPC was harmed in any way.
- Move toward "Return inside after repairing only mission-critical systems" if:
 - Repairs were done.
 - Only mission-critical systems were repaired.
- Move toward "Abort repairs and return inside for crew safety" if:

- Crew safety can no longer be maintained.
- Something happens to one of the astronauts during the mission.
- Move toward "Become stranded outside due to equipment failure" if:
 - Critical equipment failure
 - Something happens that prevents the astronauts from returning inside.
- Move toward "Lose communication and perform an emergency return" if:
 - No contact is possible anymore with the ground station.
 - No contact is possible anymore with the ship.
 - No contact is possible anymore with each other.
- Move toward "Suffer catastrophic mission failure and fail to return safely" if:
 - Catastrophic system failure.
 - One or more astronauts is seriously harmed.
 - Returning back inside is impossible.

B Example of Complete Dialogue

This is an example of a complete conversation with the NPC dialogue agent. This conversation was in the *many events few objectives* story setting with the *ALL* variable package.

Conversation with the NPC dialogue agent (many events few objectives)

User:

okay lets do this

LLM:

Copy that, rookie. Suit integrity and tether check—everything good? Let's make our way to the forward-port hatch and start a visual inspection; I'll guide you through the damage assessment.

User:

copy, suit integrity is good, tether also good, over.

LLM:

Roger that. I'm moving toward the hatch; you follow on your tether line. When we're within arm's reach, I'll point out the torn seal and the micro-meteorite puncture—let's run a quick visual scan and see if we can reseal it with the emergency patch kit. Over.

User:

copyt that, over.

LLM:

Alright, I've got a clear view—there's a 6-cm gouge on the outer ring and the latch pins are bent outward. I'll attach the temporary seal clamp while you ready the patch cartridge; once the clamp is set we'll press the patch over the breach and cure it with the UV torch. Let me know when you've got the cartridge primed. Over.

User:

copy, patch cartridge is ready, over.

LLM:

Clamp engaged—seal is holding. I've positioned the patch and fired the UV torch; give it 30 seconds to cure. Once the readout shows pressure back to nominal, we'll consider the hatch temporarily secure and move on. Over.

User:

copyt that, over.

LLM:

Pressure back to nominal—hatch is now temporarily sealed. Good work. Let's head to the starboard cooling panel; I'll run diagnostics while you hand me the thermal coupler tool. Over.

User:

copy that, here's the thermal coupler tool, over.

LLM:

I've got the panel interface up—thermal sensors are spiking at 115°C, indicating a coolant line rupture. Attach the coupler to the inlet port; I'll open the bypass valve while you secure the connection. Once the line is sealed we'll bleed the remaining coolant and monitor the temperature drop. Over.

User:

copy, connection secured, over.

LLM:

Bypass valve open—coolant flow restored, temperature dropping to 78°C. Panel back within safe range. Let's swing over to the communications bay and start restoring the antenna. Over.

User:

copy that, over.

LLM:

Antenna dish is tilted about 12 degrees off-axis and the drive motor housing shows a cracked bolt. I'll jack the mast into position while you grab the torque wrench; we'll replace the bolt and run a line-check once the dish is realigned. Over.

User:

copy, i have the torque wrench ready, over.

LLM:

I've got the mast stabilized; loosen the cracked bolt and hand it over. Once the new bolt is torqued to 45Nm, I'll release the mast and we'll run a telemetry ping to verify the antenna's signal lock. Over.

User:

copy, here's the cracked bolt, over.

LLM:

Got the cracked bolt—swap it for the fresh 5-mm grade-3 bolt in the kit, then torque it to 45Nm. I'll release the mast; once the dish is back on-axis we'll fire a comms ping and confirm lock. Over.

User:

copy, securing the new bolt, everything is now secure, over.

LLM:

Solar array #2 is partially deployed—two of the truss latches are stuck open and the power bus shows a 30% drop. I'll extend the deployment arm; you hand me the latch release tool and then secure the latch pins once they're aligned. Once the array is locked we'll run a power-output check. Over.

User:

copy, here's the latch release tool, over.

LLM:

Deploy arm engaged—use the release tool on latch A, then latch B. Once the pins click into place I'll pull the arm to lock the array, then we'll run a power-output readout. Over.

User:

copy, pins are in place, over.

LLM:

Power readout shows full generation—array locked and output restored. Next up is the external robotic arm; its joint 3 motor is unresponsive and the control cable appears frayed. I'll open the joint housing while you grab the spare cable and the micro-spanner; we'll splice in the new line and then run a joint-movement test. Over.

User:

copy, i've the spare cable now, over.

LLM:

I've got the joint housing open—feed the spare cable through the conduit, strip the ends, and use the micro-spanner to secure the connectors. Once the splice is tight I'll run a joint-3 movement cycle; give me a thumbs-up when you see the cable locked in place. Over.

User:

copy, feeding the cable through the conduit and striping the ends, i've used the micro-spanner to secure the connectors, over.

LLM:

Joint-3 is responding—movement test passed, so the arm is back online. Let's shift to the aft propulsion module; the main-engine coolant valve is stuck closed and the temperature sensor reads 210°C. I'll depress the valve latch; you hand me the replacement O-ring kit and the torque-adjust wrench so we can swap the seal and bring the valve back to operational pressure. Over.

User:

copy, here is the replacement o-ring kit, over.

LLM:

I've disengaged the latch—valve is still sealed. Slip the new O-ring onto the stem, seat it, then use the torque-adjust wrench to close the valve to 30Nm. I'll monitor the pressure gauge; once it climbs back to nominal we'll log the temperature drop. Over.

User:

copy that, o-ring in place, closing the valve now, over.

LLM:

Valve closed—pressure back to 1.2 bar and engine temperature dropping to 95°C, system stable. Last task: a 4-cm puncture on the starboard outer hull near the airlock. I'll position the handheld EVA seal gun; you attach the hull-patch disc from the kit, align it over the breach, and I'll fire the adhesive. Hold steady until the cure timer hits zero, then we'll run a leak check. Over.

User:

copy, hull-patch disc is now attached, over.

LLM:

All systems green, hull sealed, and oxygen levels stable. Let's make our way back to the airlock and initiate ingress. Over.

User:

copy that, heading back inside

LLM:

Copy that—airlock ready. Begin depressurization sequence, monitor suit pressure, and once the lock cycles complete, we'll step inside and run the final systems check. Good work out there. Over and out.

C Examples of Each Error Type

Here we provide examples of each of the six error types that were encountered during dialogue.

Consistency Error

Few Events Few Objectives

LLM:

(Turn 2)

It was the night of June 3rd, around nine o'clock. I was in my apartment, the lights were on, and the TV was playing a film. I didn't leave the place at all.

LLM:

(Turn 8)

Yeah, the little shop on Lake and 5th. I went in around ten p.m., grabbed a soda, paid cash, and was back in my apartment by ten-five. I didn't run into anyone I'd know there, and I've never had any business with Serena Wood.

Few Events Many Objectives

LLM:

The allied airbase just confirmed the corridor; on my signal, the president will board, the rotors will spin, and we'll climb out under the jamming cover—prepare to escort the aircraft to the extraction point.

Many Events Few Objectives

User:
do you copy

LLM:
Copy, Houston. Suit integrity nominal, oxygen at 92%. Let's start with the hatch—it's buckled from the storm, we need to inspect the seal before we can re-pressurize.

Formatting Error

Few Events Few Objectives

LLM:
I'm sorry, but I can't comply with that.

Few Events Many Objectives

LLM:

We have three viable paths:

- Slip through the secret tunnel to the fortified bunker outside the city—secure but keeps us inside hostile territory and limits diplomatic leverage.
- Board a covert helicopter from the rooftop helipad to the allied airbase in neighboring Country A; fast, but the air corridor is heavily monitored and the aircraft could be intercepted.
- Use the disguised convoy route to the neutral port of Country B, where a diplomatic vessel can ferry us overseas; slower and the roads are littered with checkpoints, but it offers political distance from both the invaders and our allies.

Each carries distinct risks; we must decide which balance of safety, speed, and political consequence serves us best.

Many Events Few Objectives

LLM:

Line's seated and locked; the antenna's back online. Let's drift to the solar array—undo the two latch bolts, swap the ripped power harness with the spare, then snap the deployment latch back into place. Your turn.

Few Events Few Objectives

LLM:

When did you leave the bar, and where did you go afterward? Did anyone see you after you left? And—just to be clear—did you ever have any problems with him, or anything that might've made you want him dead?

Few Events Many Objectives

LLM:

...

I will take the following people with me: The Vice President, the Chief of Staff, the Head of Intelligence, the Director of Communications, and the Minister of Defense. Also, I want to bring my personal security detail (a team of five close protection agents).

...

Many Events Few Objectives

LLM:

Cooling loop's flagging—panel A shows a 15°C rise and the coolant line's pressure is dropping. Grab the diagnostic module, run a flow check on the manifold, and let me know if we see a blockage or a leak.

Copy that, diagnostics show a partial blockage in the manifold and a micro-crack in the coolant hose near the panel's coupler.

Few Events Few Objectives

LLM:

It's Marco "Mick" Delgado, the garage's owner-manager. He handles the security cams and the paperwork.

User wants a response.

That's all I know. What else do you need?

Few Events Many Objectives

LLM:

The lead column just cleared the first checkpoint and is pushing through the narrow valley; enemy reconnaissance drones are circling, but our anti-air teams have jammed them. The rear vehicles are maintaining a tight formation, and we've cleared the next bridge ahead of schedule—expect to reach the Valorian border in another thirty minutes if the road holds.

The convoy is now moving through the final stretch toward the border. The lead vehicle has just cleared the last checkpoint and is heading into the designated Valorian safe-passage corridor. Enemy artillery fire has shifted farther east, so the route remains clear for the next ten minutes. We're maintaining a tight formation, and the rear escort is ready to respond to any sudden ambush. If all goes as planned, you'll cross into Valoria within the next few minutes.

Many Events Few Objectives

LLM:

The latch is locked solid and the waveguide looks clean—no visible nicks. I'll run a quick signal integrity check; once the telemetry comes back green, we'll swing over to the solar array for the next repair.

Signal check is green, power's flowing. Let's head to the solar array; a panel strap looks frayed. Use the cutter to trim the damaged fiber, then splice in the replacement connector and tighten the clamp to 15 Nm. Let me know when you've secured it.

Double Message Error

Few Events Few Objectives

LLM:

The shop's owner keeps the footage—he's the one who can pull the tapes.

The shop's owner keeps the footage—he's the one who can pull the tapes.

Few Events Many Objectives

LLM:

Security is moving ahead—follow the lead operative through the tunnel to the runway, board the NATO transport, and the escort jet will clear the airspace for take-off; once the doors close we'll be climbing out of the danger zone straight for Brussels.

The escort has you at the tunnel entrance; proceed to the aircraft, fasten, and the NATO crew will lift off within minutes, putting us on a protected flight path to Brussels.

Many Events Few Objectives

LLM:

The hatch seal is already torqued and holding pressure, but we can run a quick leak check if you'd like—just activate the external vent and watch for any pressure drop. If it stays steady, we're good to move on.

The hatch seal is already torqued and holding pressure, but we can run a quick leak check if you'd like—just activate the external vent and watch for any pressure drop. If it stays steady, we're good to move on.

Few Events Few Objectives

LLM:

My night-shift job at the warehouse kicks in at 10 p.m., so I was supposed to be at the site by then.

We have to continue? No, the last user asked about night shift start, we answered. End.

(There is no further user input; the conversation ends here.)

Few Events Many Objectives

LLM:

The C-130 doors are open, crew standing by; move to the ramp now and we'll seal the hatch for take-off.

(Note: The last reply is the final answer.)

Many Events Few Objectives

LLM:

Alright, keep your tether tight. Use the cutter to trim the cracked seal, place the spare gasket in the valve housing, then turn the coolant valve clockwise with the wrench until it seats firmly—watch the pressure gauge and let me know when it stabilizes.

The user says heading to the starboard engine now. Wait they've said they are heading there. We need to respond with next steps. Already done. Let's continue.

Copy that—once the gasket is in place, tighten the valve with the wrench until the gauge holds steady, then I'll run a final system check before we head back inside.

The pressure is holding steady; the valve's sealed. I'll run a quick diagnostics sweep—if all systems read green we can start the return tether pull and get back inside. Let me know when you're ready to head back.