“Wait, That Was an Option?”

Christian Huyghe

Michael VanCoppenolle

CSI 4130

4/16/2025

Abstract

This project explores the integration of large language models (LLMs) into game design with a choose-your-own-adventure game that dynamically interprets player actions and generates unique story outcomes, titled “Wait, That was an Option?”. Traditional games rely on deterministic programming, where events follow predefined sequences. In contrast, our approach leverages the nondeterministic nature of LLMs to create an unpredictable and engaging experience. Our implementation uses OpenAI’s GPT-4o-mini model to process natural language inputs and generate narrative responses. To guide the model’s behavior, we will be utilizing few-shot learning and prompt engineering techniques. Inspired by Kyle is Famous, we reference relevant documentation and research on LLM-based storytelling to refine our methodology. Since game quality is inherently subjective, we will assess our results through qualitative analysis, focusing on how well the model adheres to prompts and whether it produces compelling narratives. This project aims to highlight the potential of AI-driven storytelling in game development.

I. Introduction

Video games have been around for decades and have been a part of many important stages of electronic development. One of the latest developments, artificial intelligence, has brought a new perspective to the gaming industry. Many of the latest games rely on deterministic programming, having events tied to specific sequences of inputs from the user, with perhaps a small amount of randomization to add variety to the game. Nondeterministic algorithms such as large language models are difficult to implement in this setting, as the unpredictability of their output needs to be taken into account when handling them. Therein lies the major problem being addressed by this project, how to effectively make use of one such nondeterministic model in a way that lends itself to a compelling game.

The title of our game is “Wait, That Was an Option?”, and it is a choose-your-own-adventure game that generates story outcomes and interprets player actions via a large language model. We will be utilizing OpenAI’s gpt-4o-mini model, which is a generative pretrained transformer that takes natural language as input and returns natural language output. While the exact architecture and training data is not publicly known, the fact that it is a public model means that it does not require training datasets. In order to ensure that it returns properly formatted outputs, few-shot learning is utilized by providing the model with a few correct examples of input-output exchanges before the current input. In addition to few shot learning, a prompt is necessary to use the model, and prompt engineering has been utilized to correctly configure this prompt**.**

The result of this project is a functioning game that uses Gradio for an interface and allows for multiple-action stories guided by the user’s choices and the AI’s interpretation of those choices. The AI is able to validate the user action, check if it is consistent with the story, update the story information, and then continue the story for each choice the user makes. After a preset number of user choices, it generates a conclusion to the story that attempts to work in the predetermined end to the story while integrating the consequences of the decisions made by the user along the way.

II. Related Work

The structure of this game is largely inspired by a similar game known as “Kyle is Famous” [1], which is also a choose-your-own-adventure game that follows a well-known podcaster, as he prepares for an important interview with a prominent guest. Players guide Kyle through various choices, leading to wildly different outcomes—ranging from absurd to disastrous. This game will also involve the user making choices to progress the story towards a preset conclusion, but the difference here is that the user will not be limited in the choices that they make. “Wait, That Was an Option?” features an open input box where the user can type out any action they can think of, and the large language model works to interpret those actions within the context of the story.

Another game being used for inspiration is an online party game known as “Death by AI” [2]. This game provides its players with a scenario that would endanger the life of their player character, and they are tasked with writing out how they would survive the scenario. The AI then determines whether their solution worked, and writes out a description of how their character either narrowly escaped or met with a grisly end. “Death by AI” is the inspiration for how the validation steps for this project function, with checks being ran for action validity and consistency before determining how to continue the story, but it differs from “Wait, That Was an Option” in that it does not record any game state or continue the scenarios past the first user actions. “Wait, That Was an Option?” stores a collection of story data, and that data is updated after each action so that it can be passed to the AI in the next step.

The official documentation for the OpenAI API [3] and Gradio [4] will also greatly assist in game design and prompt engineering. OpenAI provides general tips for best practices in prompting their models, and these tips are useful for tweaking the prompt to achieve a specific outcome. The Gradio documentation includes instructions for interface design, which is helpful for creating an intuitive user interface to host the game.

III. Data

As this project makes use of a pretrained model, OpenAI’s gpt-4o-mini, it does not make use of an extensive dataset. While it is known that this model had extensive training data, the exact data used is not publicly known, though it likely includes much of the public internet. A small amount of data is used to guide the model with few shot learning, however. Few-shot learning helps in LLM applications by providing the model with an example of how to format the output. This is essential in JSON string generation, where the slightest anomaly in formatting results in a completely malformed string.

For each validation and story status update check, two to three sample input-output exchanges are provided to the model before the input, featuring the desired structure of the output. Due to the story-specific nature of the story progression steps, this cannot be applied to those specific prompts, but on the prompts they are applied to, the example exchanges have resulted in correctly formatted outputs being produced for the majority of user requests rather than happening infrequently or as part of larger outputs as they were before.

IV. Methods

The general approach that has been taken to address the problem of integrating a nondeterministic AI algorithm into a compelling game has been to divide its work into multiple tasks and give it control over the “creative” aspects of the game, essentially making the nondeterminism into a feature of the game. The AI is free to interpret the user’s actions however it chooses to as it writes the story, and it also makes the decisions on which parts of the story status to modify after each action, with error checks occurring for each output to ensure it can be read by the game backend. This is the safest way to handle the nondeterminism in this game, as the AI cannot be counted on to produce a consistently formatted output, though it succeeds most of the time.

To ensure that the AI does not make any snap decisions in its interpretation of the output, the following structure was given to the prompting calls. First, the user input is passed to a validation AI call, which checks if the response the user gave was actually an attempt to play the game. This is to prevent abuse of the AI, as large language models are known to be vulnerable to requests telling them to ignore their instructions or otherwise function other than intended. The next step is a consistency check, which determines whether the user input actually makes sense in the context of the story. While this potentially limits the full scope of what can happen within the game, it is present to keep the user from abandoning the story altogether with their prompted actions. Both of these checks are performed by the gpt-4o-mini model, and if either of them fail, the outcome generation step is instructed to make the player character fail to complete the current action or fail to act at all. If these checks pass, the AI is prompted to generate an outcome to the user’s decision within the context of the story. It is provided the rest of the story as well as a story status string to complete this task. Once this outcome has been generated, another set of calls is made to update the story status. These calls pass the user action and AI outcome to the model alongside instructions and example JSON formatting. A call is made for updating map item information, locations, characters, additions to the player inventory, and item removal from the player inventory. The outputs of these calls are all examined individually to update the story status, with checks in place to catch malformed JSON outputs. Finally, after all of these steps are completed, the user is returned the next part of the story and the new story status string. After the user has maxed out the amount of actions they are allowed to take, the AI is prompted to generate the conclusion to the story using the intended conclusion string and the story up to that point.

Each of the many AI calls has a unique prompt that was tweaked via prompt engineering. The nondeterministic nature of large language models means that there is no way to conclusively determine which prompt functions the best in all situations, but manual testing was enough to determine that the current prompts at the very least accomplish the goals they were intended to accomplish. Few shot learning was used wherever applicable to back up the prompt engineering.

V. Experiments

A variety of experiments were run for the purpose of refining the AI prompts and ensuring the output matched the intended output. These experiments involved slight trial and error prompt adjustments as well as edits to the sample input-output exchanges, but due to the nature of the model they were referencing, there was no way to assess the results other than manual assessment. The below table shows some results of the prompt adjustment experiment used to refine the outcome interpretation prompt.

Prompt Change	Fraction of valid outputs	Primary reason for failure
Telling AI to continue the story	1/5	AI fabricates additional actions beyond the user action
Telling AI not to continue past the user's action	1/5	AI fabricates additional actions beyond the user action
Telling AI to limit response to three sentences	0/5	AI fails to explain the outcome in detail
Telling AI not to repeat the user action	2/5	AI fabricates additional actions beyond the user action
Replace "continue the story" with "detail the consequences of the user action"	4/5	AI fabricates unnecessary story elements
Tell AI not to fabricate story elements	5/5	

After the AI outcome generation was made satisfactory, experiments were run to improve the AI story status update generation. While the quality of the story status updates were not scrutinized as heavily as the client-facing outcome generation, experiments did have to be run to increase the rate at which the update calls generated valid JSON strings. Below is a table with some of the results of those experiments.

Prompt Change	Fraction of valid JSON outputs
Ask for story updates in prespecified format	0/10
Included successful few-shot learning example	7/10
Include examples of empty update JSON lists	9/10
Including additional examples	19/20

While the JSON generation for the story updates is still imperfect, checks are in place to ensure that malformed JSON does not disrupt the program, so it is not too much of an issue.

VI. Conclusion

The main point that was learned from this project is that large language models in their present state are very effective at "creative" tasks such as storytelling. The model paired with the prompts that were given to it lend themselves to a relatively entertaining game that grants the user an amount of freedom that is simply not possible in other games of this type. In a standard Choose-Your-Own-Adventure game, the user can only select from a list of preset choices to pursue one of several predetermined outcomes, but the use of a large language model in this game allows the user to select any path they can imagine with the only limitation being the overall consistency of the story.

Naturally, the implementation of the game is not perfect, and could be improved upon for future projects where more time is available. Firstly, as LLMs are continuously being improved and updated, it is possible that upgrading to a newer model would result in a more consistent and possibly even more compelling game. The gpt-4o-mini model is a very fast and complex model that does not suffer from most of the major failings of early language models, so it was a good choice for this project, but there will be better choices in the future. Secondly, prompt adjustments and modifications to the few-shot learning examples might also aid in the storytelling. Instructions were given to the model in order to keep the response concise because there was an issue with it generating story details beyond what it was tasked with, but while those instructions served the purpose they were meant to, they also served to limit the full extent of the outcomes the LLM could generate. If another way was found to fix this issue, possibly by using few-shot learning to show what an adequate story response looked like, then the game would be able to feature longer LLM outputs with more story detail put into them. Thirdly, additional stories could be added to make the game more complete. This project was intended to allow for multiple story prompts, and it was coded in a way that supported having them. Due to time constraints, however, the project was restricted to only feature the story "Moe's Fishing Shack," which is a simple story intended to demonstrate the basic features of the game. If the game were to be updated, stories that are more complex than this one could be made available, expanding the scope of the game and granting an opportunity to show off what the LLM is truly capable of.

Overall, while not perfect, "Wait, That Was an Option?" is a decently entertaining game that accomplishes the task of integrating large language models into a game format. There is a large amount of room for adding new features and making the game more compelling, but that will be the task of later developers picking up this project to improve upon it.

VII. References

[1] https://store.steampowered.com/app/1186740/Kyle_is_Famous_Complete_Edition/

[2] https://deathbyai.gg/

[3] https://platform.openai.com/docs/api-reference/introduction

[4] https://www.gradio.app/docs

[5] Z. Xie, T. Cohn, and J. H. Lau, "The next chapter: A study of large language models in storytelling," *Proceedings of the 16th International Natural Language Generation Conference*, Jul. 2023. doi:10.18653/v1/2023.inlg-main.23

[6] https://www.ibm.com/think/topics/few-shot-learning

[7] http://coursera.org/articles/what-is-prompt-engineering