universität wien

# MASTERARBEIT | MASTER'S THESIS

Titel | Title

Large Language Models (LLM) for NPC interactions in a dialogue-based textadventure game

verfasst von | submitted by

Kang Da Ji BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2025

| | |
|---|---|
| Studienkennzahl lt. Studienblatt \| Degree programme code as it appears on the student record sheet: | UA 066 921 |
| Studienrichtung lt. Studienblatt \| Degree programme as it appears on the student record sheet: | Masterstudium Informatik |
| Betreut von \| Supervisor: | Univ.-Prof. Dipl.-Ing. Dr. Helmut Hlavacs |

# Acknowledgements

I would like to express my sincere gratitude to my supervisor Univ.-Prof. Dr. Helmut Hlavacs for giving me this opportunity to work on this topic and his guidance while writing this master thesis.

Furthermore, I want to thank my friends Michaela, Michael, Liana and all of my other friends who, through their support, helped make this master thesis possible .

Lastly, I am grateful to my parents, who encouraged my academic endeavors and gave me the financial stability to continue my studies over all these years.

# Abstract

For a long time non-player characters (NPC) dialogues in games were hand-written and required time and resources that could otherwise have been invested in developing other areas of the game. Due to the emergence of new technology, writing character dialogue may be substituted by Large Language Models (LLM) text generation, instead of writing the NPC texts per hand. Additionally, using models has the advantage that the game would be capable to react dynamically to player inputs and as a result, increase the interactivity with the game.

This thesis aims to research to what extend LLMs are capable to generate dialogue in a textadventure, where each character powered by Llama 2-chat models. Using Retrieval-Augmented Generation (RAG), which is a method to fetch relevant knowledge and provide them as context to the LLM, we implemented a game, in which players advance in the story by typing certain solution phrases that will let the player progress to the next character. The correct answer is determined by comparing the input with the predefined solution using the RAG model. Moreover, player's experience was recorded using a questionnaire where the performance was rated and feedback was given on each character. In total, the model received high ratings in all categories like the use of correct English, coherency, novelty or meaningfulness. Although, by analyzing the game logs and feedback we found that the model frequently hallucinates information when RAG did not find any relevant knowledge to the players' inquiry. Additionally, we have seen that sometimes correct answers were not correctly detected by the RAG system, thus incorrectly preventing the player from progressing the game.

Furthermore, we assembled a questionnaire in which we identified the preference of participants and their ability to distinguish between human-written dialogues or LLM generated dialogues. As a result, we have found that no clear preference could be determined between either option, except when the text included stuttering, in which the human-written stutter proved to be more natural. However, based on the data, we found that the preference of either text is heavily influenced by which the participants thought was the human text.

# Kurzfassung

Die Dialoge von Nicht-Spieler Charakteren, auch bekannt als NPCs, wurden für lange Zeit von menschlicher Hand verfasst, welche Aufwand und Resourcen beanspruchen, die man anderweitig verwenden könnte. Durch die Weiterentwicklung von neuen Technologien, wie zum Beispiel neuronale Netzwerke, wie Large Language Models (LLM), haben wir die Möglichkeit bekommen solche Dialoge stattdessen generieren zu lassen. Zusätzlich hat man den Vorteil, dass solche Konversationen individuell auf die Eingabe von User:innen reagieren können, welches die Interaktionsmöglichkeiten mit dem Spiel verbessern kann.

Diese Masterarbeit zielt darauf hin, ein solches System, unter Anwendung eines LLM und dessen Fähigkeit Charakterdialoge, in einem Textadventure zu analysieren. Mithilfe von Retrieval-Augmented Generation (RAG), welches eine Methode ist um relevantes Wissen zu holen das als Kontext für die Generierung verwendet wird, entwickelten wir ein Spiel, bei dem Spieler:innen Fortschritte machen, indem sie sich mit den Charakter:innen unterhalten und bestimmte Lösungssätze verwenden die ihnen den Weg zum nächsten Charakter:in weisen. Um festzustellen ob die Eingabe der Spieler:innen korrekt war, verglichen wir die Ähnlichkeit mit vordefinierten Lösungssätzen mithilfe von einem kleinen sentence-transformer Modell welches wir ebenfalls in RAG verwendet haben. Das Spielerfeedback wurde mithilfe von eines Fragebogens festgehalten in welchem Spieler:innen die Verhaltensweise der Charakter:innen bewerten und kommentieren konnten. Zusammenfassend war die Bewertung in allen Kategorien, wie die Verwendung von korrektem Englisch, Kohärenz, Neuartigkeit, Bedeutsamkeit und weiteren Kategorien, sehr positiv. Jedoch fanden wir in den Spielaufzeichnungen, dass Charakterantworten oft auf erfundenen Informationen beruhten. Dies passierte, wenn Spieler:innen fragen stellten bei dem RAG kein relevantes Wissen finden konnte. Außerdem war die Anwendung von RAG nicht immer zuverlässig, da manchmal Eingaben nicht als korrekte Lösungssätze gesehen wurden, obwohl das der Fall war.

Des Weiteren, konstruierten wir eine Umfrage bei dem wir Teilnehmer:innen zu ihrer Präferenz und zu ihrer Fähigkeit, zwischen von Menschen geschriebenen Dialogen und LLM generierten Dialogen zu unterscheiden, befragten. Wir fanden keine signifikante Präferenz zwischen von Menschenhand geschriebenen Konversationen und generierten Konversationen, ausgenommen wenn der Text stottern beinhaltete, wo das menschengeschriebene deutlich besser war und als natürlicher wahrgenommen wurde. Außerdem konnten wir anhand der Daten feststellen, dass die Präferenz der Teilnehmer:innen stark davon beeinflusst wurde, je nachdem ob sie glaubten, dass ein Mensch der Autor war oder nicht. Dies war unabhängig davon ob der Autor tatsächlich ein Mensch war oder auch nicht.

# Contents

# 1. Introduction

In many games, non-player characters, also known as NPCs, are essential for the player experience. Not only do these characters make the world more lively, they also provide important context information about the environment and progress game. Unfortunately, writing these dialogues is time consuming and costs resources that could otherwise be allocated to different aspects of the game.

With the rise of Large Language Models, also known as LLMs, developers have the ability to generate these NPC dialogues on the fly and make them responsive to player input. This strategy not only saves valuable resources, but might also enhance the players gaming experience by making the NPCs able to respond directly to any inquiry as certain in-game characters.

While much research has been conducted regarding LLM generation, the research benefit from hands-on experiments of LLMs integrated into NPCs in games. It is unclear if integrating current models, changes the user experience in comparison to human-written dialogues and if LLMs can be used as a substitute for human-written conversations.

In this master thesis, we attempt to implement such a game using an LLM called Llama 2-chat, which was specifically trained to be suited for chatbot conversations [TMS+23]. We hypothesize, that current LLMs are already advanced enough to be able to dynamically and sufficiently generate different NPC personalities. However, while LLMs can produce dialogue fast, it still relies very much on provided context and instructions. By using Retrieval-Augmented Generation (RAG), we attempt to mitigate this issue and provide the model with highly relevant pieces of documents, such as knowledge and previous dialogues, based on the user input. Furthermore, we seek to understand if users have a preference towards human-written or generated dialogues.

We aim to answer two research questions:

- **RQ1:** To what extend are Large Language Models, in combination with Retrieval-Augmented Generation, generate dialogues suitable for non-player character interactions?
- **RQ2:** Do users prefer Large Language Model or human-written dialogue and for which reasons.

We hope that the findings of this research can be used to advance the use of LLM in NPC dialogues. To achieve this, we build a textadventure game where the user progresses through the story by talking to different NPC personalities and obtaining new knowledge

that can be used in subsequent dialogues and advance through the game. The solution can be found by typing certain pre-defined phrases that convince the characters to lead the player to the next NPC. In order to determine if these phrases are used in the player input, we applied the Retrieval-Augmented Generation (RAG) concept and utilized a small sentence transformer called mixedbread-ai as a retriever to embed and compare the similarity of the input sentence and the solution phrase. Furthermore, the retriever also has the responsibility to find relevant knowledge and previous dialogues which are then provided as context for the LLM generation. To further guarantee that certain information in the generation are mentioned, we implemented trigger phrases that are activated when input and the solution phrase are similar enough. Moreover, the mechanism also constructs a fake user dialogue as context which contains the necessary knowledge and subsequently append additional instructions to the end of the user input referring to the knowledge. Using this technique we achieved a more consistent generation of necessary information in the character response.

In a within-subjects design, we evaluated the players experience remotely and in-person by letting them rate the performance of two differently sized models and collect their feedback on them. Additionally, in order to investigate if humans had certain preferences in NPC dialogue, we designed a questionnaire in which participants were presented conversation pieces and had to evaluate them on different categories and indicate which of the conversation they preferred more. Furthermore, participants had to determine which dialogue was human-written and which of them was generated by our Llama 2-chat model. The resulting data gave us insight in the decision making of the user and helped us understand how users chose their preferences.

In this master thesis, we first talk about some foundations of Large Language Models in order to understand the technical background of the technologies used in the game, such as transformers and Retrieval-Augmented Generation. Afterwards, we review related works that contain similar efforts to utilize models in NPC dialogues. Furthermore, a detailed implementation of the game is given in which we discuss the configuration, architecture and the NPCs in detail. We also discuss how we attempted to fine-tune the Llama 2-chat model using a dataset called LIGHT of real people role-playing in a fantasy setting. Next, we discuss the experiment and questionnaire design and present the gathered data, which we will then interpret in the following chapter including the limitations of our findings. In the last step, we summarize our thesis and conclude our work including possible ideas that can be improved in future research.

# 2. Related Work

## 2.1. Fine-tuning

Fine-tuning is the process of using additional data to enhance an already pre-trained LLM and focus on a special area of expertise based on the provided information. The model tends to generate text that is more similar to the newly added data while still having the vast knowledge and language skill of the original LLM. This process is also much cheaper in resources than retraining the whole model [AWK+23].

Van Stegeren and Myśliwiec [vSM21] utilized a fine-tuning method to generate new quest descriptions and titles in GPT-2 by training it on a dataset. Even though the papers main goal is to generate new quest data, it also tries to generate prior NPC dialogue that explains the questgoal and gives context on the character motivation. The training dataset was acquired from World of Warcraft quests with NPC dialogue and annotated with tags that describe datapoints like the title, objective and the description of the quest itself. The dialogue is produced by providing the fine-tuned model the title and objective and afterwards GPT-2 tries to generate a suitable description and dialogue.

The researchers let participants choose between hand-written and generated quest and dialogues. They continued to average the results and found that the quality of the text, the clarity of the text and the originality was statistically significantly worse than the handwritten ones. But the two also found that the model generated text that is on par in creativity and surprise [vSM21].

Even though this paper shows an interesting approach on how a model can generate quests and dialogues, the paper is unfortunately outdated in the sense that GPT-3 is much better than GPT-2, because it was trained on a significantly larger text corpus. The results may not be applicable to more recent LLM models which are continuously improving [vSM21]. Despite all this, the paper has valuable information about how to measure the satisfaction of users.

Another paper that uses fine-tuning is Vincent et al. [VSD+23]. Their goal was to train a completely new language model and generate dialogue that is true to a character in the sense of style. The researchers started by using already available speaker data from Cornell, a corpus of film scripts, and produced a new dataset called Cornell-Rich which is a extension of the Cornell dataset but with some metadata added that was deemed relevant in the training process. This included for example age, profession, religion, gender and a characteristic quote. Since they did not start with an already pre-trained

LLM, they decided to train it directly on the Cornell-rich dataset. This approach was insufficient in quality, at which point they decided to use a corpus with document-level information, which was way more abundant than metadata information, and treated the past dialogue of any sentence as metadata instead. This increased the quality of the models and produces data that is better in perplexity, which is the ability of predicting the next word in a sentence, than the non-contextual language model for a specific speaker. The researchers continue to state that the performance of a speaker metadata based model is comparable to one that is only using dialogue for fine-tuning. They also tested it on speakers that were not trained on in the fine-tuning phase and concluded that their approach is still performing better than the base language model, although less significant than in the previous results.

The paper of Vincent et al. [VSD+23] gives interesting insight in how fine-tuning can not only be used with past dialogue data to generate dialogues.
The use of metadata to generate new dialogue is novel and the performance is surprisingly good. The researchers show that NPC dialogues can also be generated using metadata which is far easier to specify than writing them, which shows a promising approach on generating NPCs generically. The only critique is that the paper is not clear about what document-level information is and in general could be clearer in their wording, which could lead to confusing or ambiguous assumptions.

A similar use of fine-tuning LLMs for NPCs to the one from van Stegeren and Myśliwiec [vSM21] comes from Ashby et al. [AWK+23]. In this paper the researchers also try to generate quest and dialogues using GPT-2, but additionally implement a knowledge graph for consistency. They use fine-tuning to generate quest outputs in which they should also reference the entities and objects provided in the quest description. However, the fine-tuning is only a minor point in the paper, and an extended explanation will be given in the content about memory retention.

In summary, the use of fine-tuning is an important method to improve the generation quality and generate texts that are much more similar to the ones provided to them.

## 2.2. Role-playing

Role-playing is a concept in which the prompter asks the LLM to impersonate a role that was given. It is often used in papers that try to emulate specific characters in fiction or letting a model know to behave in a certain way [SMR23]. This approach saves time compared to fine-tuning, as it is much simpler to assemble a prompt than to locate the data and train the language model for hours or days, with similar performance as fine-tuning [MLWF21]. Using this technique one can also correct flaws and change properties of the character on the go, without much effort or knowledge about training or modifying the model. But even though role-playing is an easy way to get the desired output, some papers use this technique in combination with fine-tuning.This allows better quality outputs that fits more to the application area.

One paper that is using role-playing is Li et al. [LHI+23] which proposes an "AI Society" of two LLMs with assigned roles that perform their respective task and have conversations with each others. The two roles in the papers are AI Assistant which is a Python programmer and a AI User, which is a stock trader. Aditionally, a third LLM refines the human user input into an improved prompt in order to design more specific tasks for the AI User and Assistant to follow. The AI user provides the AI Assistant with new instructions and the AI Assistant will respond with an appropriate solution that completes the task. This process will be utilized as the past dialogue and the AI User will, again, begin the instruction-giving, but this time with additional past dialogue.

Additionally, Li et al. [LHI+23] also suggests a component that acts as an critic, which is either a human or another AI agent to steer the conversation, by selecting and providing feedback to the proposed solutions in-between the loops. The conversation stops if it reaches a certain amount of messages, tokens or one of the other termination conditions. For evaluation the researchers use human evaluation as well as a fine-tuned GPT4 evaluation. They come to the result that using their kind of approach comes to a better solution than a standard one prompt gpt-3.5-turbo output.

Even though this type of prompting is useful, Li et al. [LHI+23] observed that in a conversation between two LLMs, an agent switched roles from posing the question to answering them, when the other agent transitioned to asking questions instead of answering them, ignoring the assigned role. This caused both agents to perform a "role flip". Li et al. suggests that to prevent this behavior, one has to prevent the LLM from switching to asking questions.

Similar behavior might happen when the user is allowed to converse with the NPC freely, which might steer the character away from the intended personality. This suggests that some mechanism is needed, where the user cannot ask certain questions that would change the role of a character [LHI+23].

The paper of Li et al. [LHI+23] provides us with valuable insights about possible problems in role-playing characters which needs to be addressed in future works. It also includes useful techniques to improve the quality of human prompts using another LLM. Unfortunately, the paper lacks a clear evaluation metric since the scoring of the output is either done by subjective human scoring or an intransparent GPT4 model. The evaluation is unclear and intransparent and would benefit from a set of prepared questions or guidelines.

Other researchers also called Li et al. [LLY+23] propose a role-playing and fine-tuning tactic method to emulate specific screen characters. The paper states that the basic ChatGPT model is already capable of generating basic functionality, but criticized that the personality it was not convincing enough and it was not repeating classic sentences from their respective mediums. The researchers claim that using role playing relies heavily on the memory of the LLM itself. If not enough information is already trained or insufficient information is provided, the model will not act true to the requested character. This is why they implemented a system to search original dialogue of the character using

a sentence embedding model and use it on the most similar original character sentences, using cosine similarity, to find the reply that fits most to the question [LLY+23].

On the other hand Li et al. [LLY+23] also used fine-tuning the model additionally to role-playing, but according to them this caused higher incorrect statements, which are called hallucinations. Although hallucinated statements may appear accurate, they are contradicted by actual data.

Furthermore, the researchers stated, that some characters lacked training data. They proposed an approach where additional data for a specific character was generated by simply feeding existing character dialogue to a Large Language Model and let the model continue the scripted conversation with its own generated dialogues. This generated dialogue would then be used as training data for the Large Language Model.

The paper Li et al. [LLY+23] is an interesting but not surprising approach to emulate characters using an LLM. The most interesting knowledge lies in the dialogue generation for training when there is too little information to be found. Although, it may be questionable to use this data for training without human supervision, since it can generate completely uncharacteristic dialogues.

Li et al. [LLY+23] also did not translate a few of the dialogues appearing in figures, making it unreadable for non Chinese speakers. Otherwise the paper still has a pending evaluation and only provides a table of outputs of the various models they want to compare. Unfortunately, the paper has no way to evaluate the performance of their models objectively in any way besides these few example outputs, which renders the paper incomplete.

## 2.3. Memory Retention Methods

A major point in NPCs is that they know their own background story and have knowledge of the world they live in. These pieces of information have to be accurate and in character but LLM outputs are sometimes incorrect, contradicting or repeating themselves semantically [BMR+20]. This is an issue when players ask NPCs about the lore of the world or personal questions that may contradict previous knowledge or do not give any useful information in that context.

In Tsai et al. [TZL+23] the researchers attempted to test the memory capabilities by playing the textadventure game Zork, using ChatGPT. The game is playable purely with text input and allows a clear set of steps to get to the goal of the game. Using this the researchers tested their system by explaining the game principle and insert the LLM the game text along with all the possible game choices. Tsai et al. [TZL+23] fed the LLM the layout of the game map by playing the game and asking it to remember the rooms. ChatGPT gave correct instructions on how to go to neighboring locations (one step), but failed at locations that were farther away than one (multi step). It also seemed to have knowledge about the game, probably from an online walkthrough of the game, and

was even able to play some of it correctly, until it hallucinated new objects that were not part of the solution. Additionally, they also discovered that ChatGPT often forgot previously stated actions, which lead to confusion. But this was mitigated by asking it to remember chosen actions.

Tsai et al. [TZL+23] also discovered that ChatGPT sometimes refused to answer the question or mimicked a human player when it did not know what to say. Similar behavior was also observed by Li et al. [LHI+23] when two LLM agents were talking to each other, where the agent repeated the instructions of the question giver. This behaviour is problematic because it can lead to a dead end in the conversation by repeatedly answering the question semantically the same, providing no new information to the player [TZL+23].

Even though Tsai et al. [TZL+23] do not implement a novel system themselves, their investigation about the topic is useful in discovering possible problems in memorization of LLMs, even with ones that are very well trained. Their research raises the question if LLMs are able to fulfill larger player requests such as asking for directions or recounting important information about the world and themselves accurately. Unfortunately, this is the entirety of their research findings.

A possible mechanism to introduce some coherent world knowledge to the NPC is implementing RAG by using a database or knowledge graph to represent entities, locations and relationships between them, as Ashby et al. [AWK+23] have done in their paper. did within their paper. The researchers goal was to generate quest descriptions and NPC dialogues for RPG games using GPT-2. The quest is generated by looking up the entity that was asked and use that as the starting node and determining which action the players intention when they speak to the NPC. The second step is to collect all relationships that contains the action and involves the starting node and convert it to the English language. Afterwards it is further processed to a vector representation of that sentence. This enables the system to identify the most probable information by employing cosine similarity between the question and the vector representation, selecting the vector with the highest score. These nodes are then followed and stored to generate the final quest description. Using this quest description the researchers are then able to feed it to the fine-tuned language model, which is trained on dialogue generation that has the quest description. The model then generates NPC dialogue that can accurately respond with information and objects in the world. [AWK+23]

Ashby et al. [AWK+23] evaluated the performance by asking online participants to rate the quality of their application, 4 other models that are all based on fine-tuned GPT-2 and hand-coded quests from World of Warcraft. The questions were designed after a modified method from van Stegren and Myśliwiec [vSM21] consisting of 4 categories. Ashby et al. [AWK+23] also deployed a metric for dialogue relevance that includes measuring the occurrence of quest description words in the dialogue and normalize it by dialogue length.

The results that they found is that even though the quest description and the dialogue did not really fit to the question the user inputted, the participants still chose the hand-written quests and dialogues over the generated ones. This was largely because

players found the output semantically incongruent and also started NPC interactions with non-related specific information like “Do you have any quest?”. Although, even with this approach the researchers report that the language model sometimes invents additional non existent information and the performance of the output is largely affected by the extent of the knowledge graph or rarely does not produce any dialogue at all [AWK+23].

Overall the paper of Ashby et al. [AWK+23] gave us an idea on how to generate more consistent outputs that hold true to the game world and consist of actual entities and locations in the world. On the other hand, this approach takes a lot of effort to make, since a knowledge graph about the whole world and which has to be updated dynamically continuously is not very applicable to a large video game with thousands of locations and entities. Furthermore, the results and inconsistency put into this approach is not good enough compared to the resources invested in it. But while this is true, the idea is not bad and future work might find better solutions to reduce cost and performance.

In the previous section we focused on creating roles using well defined roles and prompts. The project from Shao et al. [SLDQ23] approached the method of role-playing differently by using the past experience of a character to simulate them using the LLaMA language model. The researchers called this experience upload, which goal is to mimic the mental and physical behaviour of existing characters instead of duplicating only their style and tone. In this endeavor they built a dataset consisting of a profile, a scene and interaction. The Profile describe the personality and a comprehensive introduction of the characters experiences from early childhood to the most recent events. On the other hand the scene describes the environment around the character in a specific interaction. Lastly, interaction describes the cognitive process and utterances of this person.

Shao et al. [SLDQ23] mention that LLMs produce hallucinations or have knowledge outside of their temporal or spatial environment, like knowing about smartphones in the Renaissance. This is why they additionally propose to include protective experiences which is a method to fine-tune the models using “protective scenes” which are conversations from a very inquisitive person that proposes questions that constantly contradicts the characters identity. This lead the character to feign ignorance or be confused when being asked questions out of their knowledge realm.

The evaluation was done by generating questions across different topics using ChatGPT and use GPT-3.5 to act as a judge across 5 categories. Memorization which covers correct information recall. Values, which involves having the same objective and values. Personality, which should mimic the characters speaking style and emotions. Hallucination, which rates the lack of knowledge of a character about things it cannot possibly know and expresses them accordingly. Stability, which investigates the acting and consistency during longer periods of dialogues [SLDQ23].

Shao et al. [SLDQ23] concluded that they successfully reduced hallucinations and outperformed other LLMs like Alpaca 7B and Vicuna 7B by a large margin. They have also found that their model expressed similar performance as the ones from ChatGPT with a

much smaller scale (7B).

The research and idea by Shao et al. [SLDQ23] is an exciting approach to the topic of LLMs in NPCs. Not only is the idea of using experience as an information source beneficial to generate accurate data, but Shao et al. also introduced a way to reduce hallucinations, unlikely knowledge and possibly malicious attempts to influence the character.

The downside of this approach is that it requires a lot of detailed background information not only of the person but also of the environment in which the interactions happen. In the context of NPCs this means that this is an extremely time consuming endeavor, which might not be applicable to all NPC characters in a game. Also the judging may be inaccurate for ChatGPT since GPT-3.5 is a successor of ChatGPT and might have an implicit bias to favor texts that are more like its own outputs. Nonetheless, their research is very valuable in future character creation especially in the area of accurate memory depiction.

## 2.4. Evaluation Strategies

The output quality of an LLM is largely based on subjective opinion. This makes it tedious to gather uniform data about the quality of dialogues. Evaluation of the dialogue quality requires a clear set of questions that describes an enticing and lively conversation with a NPC.

Van Stegeren and Myśliwiec [vSM21] provide such a metric based on language quality, coherence and creativity. With quality the researchers meant if the generated text was using correct English, while coherence was asking about the semantic understandability of the output. Creativity was harder to define, but they used Boden's breakdown of creativity [Bod09] to formulate three questions that asks about novelty, surprise and value. But because there was confusion about the "value" term, the reserachers decided to change it into "creativity" instead. [vSM21] Van Stegeren and Myśliwiec then proceeded by computing the average of each property to assess the performance in each respective category.

Other papers in similar areas based their quality measurement as for instance on the paper of van Stegren and Myśliwiec [vSM21] but adjusted it because user feedback indicated unclear definitions of the terms. Ashby et al. [AWK+23] found that the term "Surprise" had multiple interpretations and decided to remove it completely from the questions. Csepregi [Cse23] also adopted this measurement, but added options to reason why the outputs were rated that way.

Xu et al. [XGDM23] and Li et al. [LHI+23] decided to utilize GPT-4 to score and pick the better performing model in comparison. Although, Li et al. also uses human scoring where they evaluate which model performed better than the other or if they were equally

capable. But also Shao et al. [SLDQ23] used LLMs, namely ChatGPT and GPT-3.5, for generating interview questions and also scoring of the performance of various models.

In summary, not a lot of evaluation strategies exists for evaluating NPC dialogues with LLMs. The most promising method is the metric from Van Stegeren et al. [vSM21] since multiple papers have used it in their research. However, some researchers decided to use LLM models to compare the results and make a more automated way to measure their performance.

# 3. Foundations of Large Language Models

In the past few years neural networks has risen to popularity because of its versatile application. One of the most popular architecture at the moment is the Large Language Model (LLM). A language model is defined as probability distributions over words, symbols and tokens [TLI+23] [Sha51]. This means that it has the ability to statistically predict the continuation of a sequence of symbols by evaluating the conditional probability of the next word based on the available previous words.

## 3.1. Large Language Models

As their name implies, Large Language Models are trained on vast amounts of data to make a more accurate prediction [JM25]. The reason why Large Language Models are so popular is their wide application in many areas which normally need human workforce for mundane, but time consuming tasks. Text generation, such as summaries of texts or code generation, are often used to ease up the workload and can generate large bodies of texts.

Furthermore, LLMs are able to parallelize their training instead of having sequential training as in recurrent neural network models [VSP+23], which makes training on large sequence data possible and more efficient. Using LLMs it is also possible and more efficient for making bigger, more scalable models than neural network models as shown in Kaplan et al. [KMH+20]. However, the inner machinations of LLM are complex, so to help unfamiliar readers to understand this thesis better, we elaborate on the basic mechanism of LLMs in this section.

The transformer architecture is the basis of LLMs, while an LLM is the term to describe a model that is already trained on vast amounts of data using transformers. To explain the architecture, we refer to the original paper of Vaswani et al. [VSP+23], who introduced the concept of transformers for the first time. An illustration of the architecture can also be found in Figure 3.1. In the paper, a key concept of the transformer is a mechanism called self-attention, which allows the model to learn the contextual significance to other words in the same text. As a result, this process allows the model to focus on the more relevant parts of a text using one attention head. Multiple of these attention heads are present in each attention layer and are trained separately to detect different types of relevancy [VSP+23]. First of all, transformers consist of roughly two parts. Multiple stacked encoders, which convert the input text into an internal representation

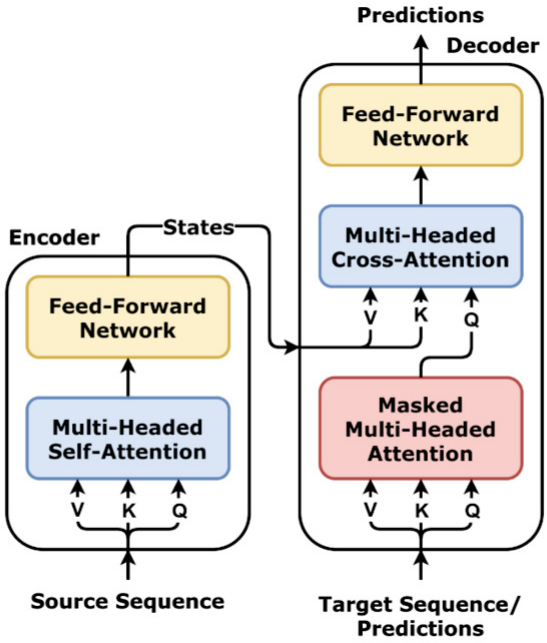


Figure 3.1.: Encoder-decoder block of the transformer architecture [dvg25].

and multiple stacked decoders which use the internal representation to generate the most likely answer to the input [VSP+23] [JM25].

In the first encoder which is also the bottom of the stack of all other encoders, the user input is transformed into a vector representation, using an embedding algorithm. Each encoder and decoder has one self-attention layer and one feed forward layer. The self-attention layer has the task of comparing and evaluating the likelihood and in turn relevancy of each word to all words in the sequence and producing a vector. This is done by producing three different types of vectors out of the vector representation by using three trained weight matrices, “value”, “key” and “query”. Each word has their own three vectors and computing the relevancy to all other words is done by using the dot product of one word and the key vector of all key vectors including its own. This is done with every word in the text until we have a score for each of the words, note that this is still the process of computing the relevancy of a single word. We now have a score how similar each word is to all words in the sentence [VSP+23]. Then this score is divided by the square root of the dimension of the key vector, in the case of Vaswani et al. [VSP+23] it was 64 so they divided it by 8, to get more stable gradients. We then take the softmax of the value and multiply it with their respective value vector in order to make unimportant words less relevant. In the last step, all of the value vectors are summed up and as a result we have the vector representation of a single word. Subsequently, these operations are done to every single word in the text. This calculation is called attention head and the layer uses multiple of these heads, which have different weights and thus have different

types of results, to generate multiple outputs that are passed along to the feed forward layer. The reason multiple heads are used, is that each attention head evaluate each word differently. As a result, we end up with multiple interpretations on which words are relevant to each word in the sentence [VSP+23]. As a result, since the feed forward layer requires a single vector representation, all resulting vectors are concatenated before getting passed to it. The feed forward layer applies further computation on the vector in parallel and continues to pass it on to the next encoder and the process is repeated until no encoder is left [VSP+23].

On the other hand, the decoder consists of roughly the same components as the encoder with the difference, that it also contains an additional layer called Multi-Headed Cross-Attention. The first decoder receives a "Start of sequence" token as input and passes it to the self-attention layer like in the encoder. However, after processing it, the output gets passed to the Multi-Headed Cross-Attention layer, which combines their own input with the output of the last encoder, giving higher weight to parts of the encoder output that are more relevant for the generation. Furthermore, after each decoder computed the information, always using the encoder output as the basis, the resulting vectors are then processed in the linear layer. The linear layer responsibility is to turn the information into a logits vector with scores. Here, each cell position is the representation of a single word and the value represents the score of one of the known words in the system. Consequently, the size of one logits vector shows how many words are known to the model. [VSP+23]

Finally, this vector is then processed by the last layer called the softmax layer, which turns the vector scores into a distribution of probabilities that adds up to 1. Now, we can determine the word that is most likely to occur next by finding the word with the highest probability in the vector. As a result, the transformer now has the next word in the sequence which will be used in the next iteration of the decoder until an "End of sequence" token is generated, signaling the end of the generation. This process of procedurally generating the output based on the previous sequence is called autoregressive generation [VSP+23].

In summary, a transformer determines the relevancy of each word to all other words in a given text. Each transformer has encoders, which transforms input text into an internal representation and passes this information along to the decoders, which procedurally predicts which known word most likely will occur next in the output sequence until it creates the complete text [VSP+23].

## 3.2. Fine-tuning

An LLM can be utilized in many different types of applications, but sometimes they require more specific knowledge. For example, code generation might already be included in the initial training data of the model, however it is possible that it has never seen a specific programming language and thus struggles to generate the correct code for that language. In this case, the solution is to provide it many code snippets of that language

and fine-tune it on that data. Generally, fine-tuning is the process of re-training a model, by using new data to change the behavior of the LLM to the specified information [JM25]. The model is trained by using stochastic gradient descent, just like in other neural networks, in which the goal is to minimize the loss function over the training data, which is split into smaller sub-sets called batches. This process updates the weights and permanently changes the behavior of the model to be able to fulfill specialized requests [JM25].

The difficult task is to find sufficiently large amounts of good data to feed it to the training so it can accurately learn from it. Additionally, re-training the entire neural networks, without other mechanisms, could lead to performance issues in other abilities, since the focus of the model shifts to operate on other tasks, which is also called catastrophic forgetting [Fre99]. Having the correct pre-trained model is also important to increase efficiency and accuracy. A pre-trained model has already run through training on data which results in an LLM and is the basis for fine-tuning [JM25]. For example, a model that already has been trained to generate code, should perform better and minimize the loss quicker than a general purpose LLM.

In LLM, instruction following is especially important in chatbot applications like ChatGPT [ZSW+20], which popularized the LLMs in the general public. To improve the ability to follow instructions in an LLM, Wei et al. [WBZ+22] introduced the concept of instruction tuning. In this fine-tuning technique, before training, the dataset is modified so that it contains an instruction in the prompt, which serves as the context for the model, to each datapoint and further information that helps fulfill the instruction. Additionally, each datapoint also contains the desired response to the instruction so that the behavior of instruction following can be learned by the LLM. As a result, trained models can follow instructions from users in the form of natural language like “translate this text” or “ summarize this text” and other commands much better [WBZ+22].

## 3.3. Low-Rank Adaption

Even with pre-training, fine-tuning models requires very good hardware and larger models require even more resources which make them very costly to train. The problem is, that the model needs to be loaded into the memory in order for it to be updated after each batch iteration. Low-Rank Adaption (LoRA), which was introduced by Hu et al. [HSW+21], mitigates this issue by only tracking the changes in a separate weight matrix, which only has to be added after the training is done. This weight matrix has the same dimension as the original one, however LoRA uses an additional trick to reduce the memory load. By using matrix decomposition, the matrix is split into two matrices, called update matrices. The matrix can later be reconstructed by multiplying the two decomposed matrix with each other. Although this process does loose a bit of accuracy, depending on the dimensions. The other dimension, which is called the rank, of the matrix can be changed to a higher rank to achieve better accuracy. The bigger the matrix

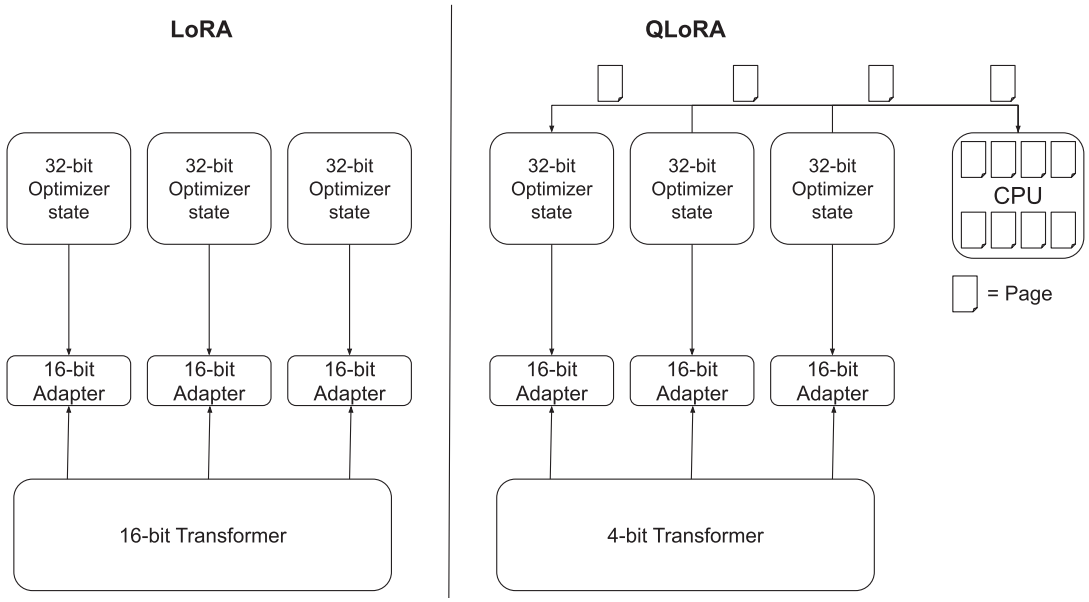


Figure 3.2.: LoRA and QLoRA architecture based on the paper of Dettmers et al. [HSW+21].

dimension is, the more memory is saved by using LoRA. At the end, the weight matrix is applied to the original model weights using an adapter and get a new fine-tuned model [HSW+21].

If this memory reduction is still not enough, one can achieve even more reduction using Quantized Low-Rank Adaption (QLoRA). Here, Dettmers et al. [DPHZ23] found a way to turn 16-bit weights into 4-bits quantized weights in a custom datatype called 4-bit NormalFloat. Using this datatype the weights can be reconstructed again using a zero-centered normal distribution. The quantization constants can then be further quantized in a process called Double Quantization. Additionally, to handle any out of memory errors while training on a single GPU, the researchers implemented a method where they swap out allocated paged memory between the CPU and GPU memory and load it back in when the GPU requires it again. This prevented any memory spike problems while training. By using these techniques, QLoRA reached similar performance as ChatGPT 3.5-Turbo, while only training it 24 hours [DPHZ23]. Both LoRA and QLoRA architectures are illustrated in the Figure 3.2.

## 3.4. Llama

One popular LLM is the Llama architecture from Meta. Their model is based on the decoder only transformer architecture that we described in Section 3.1 with some additional improvements. In their original research paper Touvron et al.[TLI+23],

introduced the Llama model, to show that it was possible to make a model that was trained solely on publicly available data. Additionally, the model that they have trained, can be requested by the public for private or research use. Llama is a general-purpose foundation model that is not specialized in a single topic or tasks [TLI+23]. If there is a need for a specialized model, further fine-tuning these foundational models can be used. The major changes that they implemented was to normalize the input instead of the output to increase the training stability, in a process called pre-normalization. Furthermore, the paper mentions using Swish GLU (SwiGLU) [Sha20] instead of rectified linear units (ReLU) [Aga19] as the activation function in their feed forward layers to improve performance.

Lastly, instead of absolute positional embeddings, they used rotary positional embeddings at each layer to make the calculations more efficient. Absolute positional embeddings are vectors from words that also contain the position of the word in the sentence, since the order of the words may change the meaning. On the other hand, the rotary embedding rotates the input vectors instead of their absolute position in the text to signal their relation with other words [SLP+23].

As a result, these changes managed to improve the training efficiency and at the time of the release, the model managed to compete with other state-of-the art models, for example GPT-3 [BMR+20], while they were approximately 10 times smaller than other comparable models [TLI+23]. Furthermore, they improved their initial model, publishing Llama 2 and their fine-tuned Llama 2-chat. In Llama 2, they increased the pre-training corpus by 40% compared to their first model. Additionally, the context window was doubled, which allowed the model to understand extra information and instructions by having longer prompts as context [TMS+23].

The researchers especially pointed out their Llama 2-chat model which differs from Llama 2, because it was specifically fine-tuned to handle multi-turn conversations. Here, they state that they were using Reinforcement Learning with Human Feedback (RLHF) [ZSW+20]. This reinforcement learning uses human feedback to improve the training of their models, by collecting data in which human annotators write prompts and then proceed to select which output out of two models they favor [TMS+23].

Subsequently, designing their own efficient reward function becomes obsolete and afterwards the data is used to train the model, which improves the instruction following and aligns it more to human preferences. The data was gathered weekly to keep it from overfitting and each week new preference data is introduced to maintain reward accuracy. Additionally, the researchers also included open-source datasets and combined them with their own preference datasets to prevent reward hacking, where the LLM finds loopholes in the reward system [TMS+23].

Finally, the researchers introduced the concept of Ghost Attention, in which the purpose is to teach the model to adhere to the instructions on how to behave given by the user and not forget the initial instruction in subsequent inputs. At first, the instruction is appended before every user prompt, so the model generates multiple texts that follows

the instructions, then using RLHF users pick the best response. This answer will in turn be used for further fine-tuning, however the instruction which was used in the generation is removed from the user text except the first input. As a result, the model should be trained to continuously follow the instruction that was given in the first input [TMS+23].

## 3.5. Retrieval-Augmented Generation

Another important concept is that the LLM is able to generate factual responses to user inquiries. Models have already been trained on vast amounts of data and are able to answer some questions correctly. However, re-training the model to be able to answer specialized subjects is expensive.

Here, Retrieval-Augmented Generation (RAG) comes to play, which was introduced by Lewis et al. [LPP+21]. Their RAG supports the generation of a model by reviewing and picking the most relevant external knowledge and adding them to the LLM as factual context. By using this method, researchers can extend, update and review the knowledge of models without time consuming training. Additionally, Lewis et al. [LPP+21] found that RAG can be used in tandem with the internal knowledge of the model, by using the external knowledge to guide the generation and draw out information that is already trained in.

To understand what RAG does, we first need to understand document embedding. Document embedding is an operation that takes a sequence of text and transforms it into a vector space representation of the model based on the word distribution of texts [JM25].

By utilizing the cosine similarity, we can calculate the angle between two vectors and determine how much they point into the same direction. Two vectors that point exactly into the same direction in this space means, that the representations are identical in this vector space. Subsequently, the more two vectors point into different directions, the more these text differ from each other. This method can also be used on words or other concepts as long as the models have learned how to represent them accurately in a vector space. Embeddings can be generated with encoder-only transformer models, such as Bidirectional encoder representations from transformers (BERT) [DCLT19] which was used in the original RAG paper from Lewis et al. [LPP+21] or even with non-neural network means like Bag of Words or TF-IDF [JM25].

The architecture of the RAG model consists of a retriever and a generator. Inside the retriever there are two encoder models, in their case, a BERT-based document encoder and a BERT-based query encoder. As their name implies, the document encoder embeds documents into a vector space representation, while the query encoder does the same for the user input. To increase efficiency, all documents are pre-indexed by the document encoder in the document index in order to be fetched quickly without re-encoding them every time [LPP+21].

First of all, the user input is embedded and using Maximum Inner Product Search (MIPS) operation to find several related documents to the query. MIPS tries to maximize the dot product of the query and each available document and attempts to find the one that maximizes the equation. The higher the dot product is, the more two vectors are pointing in the same direction, which tells us their similarity. After determining the similarity of all documents, we can rank their relevancy to the input, by sorting their scores. This gives us a list of documents ranked from most relevant to least relevant and by using a top-k approach, where k is the number documents requested, the model can get the most important k pieces of documents, which are in turn fed to the generator as context. Here, the generator is a type of transformer, which was a BERT model in the paper [LPP+21].

Optionally, one can re-rank the documents in an additional step using more resource intensive models or computations, however, while this approach may result in slightly better results at the cost of computing time, the results of Lewis et al. [LPP+21] suggests that even without re-ranking, RAG still is able to perform state-of-the-art performance comparable to re-ranking strategies at that time. So if additional computation time are acceptable as a trade of for a better accuracy, re-ranking the retrieved results from RAG can possibly improve the generation further.

Additionally, in the paper they introduce RAG-Sequence and RAG-Token as two different RAG concepts. RAG-Sequence finds the relevant document to the query once and then uses this as the source for the whole generation, while RAG-Token re-evaluates and consults multiple documents at each token. This means that RAG-Token can have different documents as contexts [LPP+21]. At the end, the researchers found, that users preferred the factual answers of their RAG model, over other language models at that time, like BERT.

In conclusion, the use of RAG to gather relevant and accurate information is a valuable tool for text generation.

# 4. Creating NPC Dialogues in a Game

In this chapter we will discuss how we integrated a Large Language Model as a NPC dialogue generator into a textadventure game. We will further explain how we utilized RAG to help the LLM answer more correctly, the fine-tuning we attempted and talk about the problems that arose during the implementation of the game.

## 4.1. Implementation

Since LLMs require high amounts of computation and VRAM, one or multiple GPUs were necessary to execute Llama 2-chat, depending on the parameter size of the model. Therefore, to run the models, we utilized two NVIDIA A100 GPUs, each equipped with 40 GB memory, and ran 13B and 7B parameter model. Additionally, each environment was equipped with two Intel 4214R CPUs.

However, it was not necessary to use both GPUs to execute the models. We have found that using a single A100 GPU, only about 30 GB were required to execute the 13B model and generate text, while the 7B model required only 16 GB of memory when generating. Although, if multiple GPUs are present, the memory load can be split up between them. Systems with less available memory, can be further reduced by loading the models as 8-bit weights. However, this can change the performance of the models, which is why we decided to leave out this configuration. Having sufficient memory space is absolutely necessary to execute the model, otherwise the program will throw an out of memory error. We have also not run into any performance issues while generating, unless anyone else was simultaneously using the GPU for computation. Although, this did not harm the generation, it did slow down the generation speed significantly.

Furthermore, to be able to fine-tune the model using QLoRA, the weights and tokenizers had to be converted from the Meta format to the huggingface format using a huggingface python script [1]. We decided to keep using the format because AutoModelForCausalLM library can use a variety of huggingface models and is being maintained. This gives future researchers the option to change the model more easily than using a code that is specifically designed for Llama. Moreover, AutoModelForCausalLM has a variety of parameter options both in initializing the model and in model execution that can further tweak the performance. We initialized the models with the "use_cache" option as true

[1] https://github.com/huggingface/ttransformers/blob/main/src/transformers/models/llama/convert_llama_weights_to_hf.py

so already generated tokens are not recomputed in subsequent tokens. As a result, we utilized the huggingface format in all of our experiments. To run the models we employed a temperature of 0.8 using top-p 0.9 and a max sequence length of 4000 tokens and a repetition penalty of 1.2 for both the 7B and the 13B models.

Because we want to explore the performance of LLMs, the game had to allow the player to speak with the model using a simplistic UI interface. Furthermore, the goal was to let the participants converse with multiple NPCs with different personalities and progress in the story, for instance gathering information and finishing quests that push the player closer to a goal. We decided to implement a game that functions like a textadventure to allow us to evaluate the textual quality without any visual distractions, with the added benefit of it only focusing on the conversation text without any visual distractions. Upon further consideration, we chose to implement a detective style story where players chat with NPCs and by asking the right questions, find clues on which character to interrogate next. With such a game design players get to experience the authenticity of a non-player character and their ability to hold a clear and coherent conversation in-character.

Since LLMs do not always act as intended we decided to include a solution command that automatically solves the current riddle and lets the user continue the game with another NPC. This is done to ensure that players do not get stuck and to prevent frustrations while playing the game. At the end, to give the players a satisfying experience we implemented an ending that resolves the main objective by letting the player decide who to name the culprit of the murder.

First of all, we chose the programming language Python, since many LLM libraries are based on it. Additionally, we decided to use a database in sqlite3 [Hip20] to save the game progress and inserted prompts and knowledge about the in-game world. This decision was made, because sqlite does not require an additional database management system to run.

For the implementation of RAG, we used mixedbread-ai [LSKL24] with sentence embedding capabilities, specifically mxbai-embed-large-v1, as a Retriever to rank and find texts relevant to the user input. We decided to use mixedbread-ai, because at the time of the development, this transformer was rated highly in the “massive benchmark for measuring the performance of text embedding models” (MTEB) leaderboard on huggingface [ECK+25] [LSKL24], while still being a relative small model.

There were a few models to choose from, but Llama 2 stuck out because it was a well known LLM. It also offered a license that granted us the ability to use the model for research, as long as we did not exceed a specified number of users and did not use the model output to train our own model. Llama 2 also provides a fine-tuned version called Llama 2-chat which was specifically designed to be used in the context of chatbots [TMS+23], which is the reason we decided to select it in our final decision.

In total, the architecture consists of the Controller, the Retriever, the Generator, the Database Manager, the Monitor and the Game Initializer. The program can be started

with an optional starting parameter that either accepts the letter “A” for the 13B model or “B” for the 7B model in order to evaluate both models more easily.

The most essential module is the central Controller, which controls the game logic and instructs the other models to execute operations. In addition to processing the input and displaying the output, the Controller also initializes the other modules, constructs prompts for the Generator and checks triggers based on the information from the Retriever.

The model is also loaded within the Controller, where different types of parameters like temperature, the max sequence length and the the repetition penalty can be set. The temperature changes the text generation the most, since it controls the randomness of the generated text. The range of the temperature lies be between zero and 1, where zero is very static and predictable and one is very random. The max sequence length limits the tokens to not exceed the trained range and the repetition penalty is used to prevent repeating words in the generation. Furthermore, since the model can be split up into multiple GPUs, a broadcasting system is necessary in case the models are distributed between them. Here, we used the distribution package from the library torch, to pass information from the first GPU to all others. This is important since each GPU starts their own process of the program which would otherwise require every input to be inserted twice into the terminal. In order to reduce redundancy and synchronize the processes, broadcasting will only be done by the first rank GPU. This means that the first GPU is computing every operation, like database operations, finding relevant documents and printing the generated text. The other GPUs are only relevant when they help generate the text.

Using the knowledge from RAG [AWK+23][LPP+21], the Retriever has the task to utilize the mixedbread-ai model, which is a sentence transformer, in order to find the most similar information based on an input text. It is able to find the x-most relevant information, where x is the number of entries it should retrieve, based on the queried texts from the database. To find any related texts the model requires the query prompt “Represent this sentence for searching relevant passages:” followed by the input text of the player. Then the query and all other documents are encoded into a vector representation by the model. Afterwards all documents can be compared to the input by using cosine similarity on the input and each document, resulting into a value between zero and one that signals their similarity, where zero is no resemblance and one is the same sentence. In our case, documents can be previous dialogues, character knowledge and trigger phrases. Using this method, we can fetch in-game facts and rank a given list of documents by their relevancy, find relevant conversations in the dialogue history or possibly activate any triggers. Furthermore, the Database Manager executes operations on the database. Tasks in this module include creating and fetching NPC information, triggers that are used to activate information, knowledge and past conversations. The Database Manager also closely works together with the token manager which is the tokenizer of the model and converts all knowledge and dialogues in the database into tokens. The resulting operations are saved in a sqlite3 file which is named after the session name so we are able to load the old game state. The task of the Monitor is to record all interactions

with the user and the Generator into a .txt file, which is saved in the folder “logs” and is also named after the session name. This module saves anything that the Controller tells it to do, which includes input, Generator responses and background information, which include information that was used to construct the prompt and the similarity score of the input and trigger phrases. The only purpose of the Game Initializer is to populate the database with entries about NPCs, knowledge, trigger and secrets, which are solution phrases that progress the story. To ensure that players do not accidentally stumble upon information that includes hidden knowledge, we implemented secrets to ensure that certain information is only released when the player has convinced the NPCs.

Both triggers and secrets are used to compare the input to their entry by using the Retriever to see if their similarity is above a certain threshold. Each secret has a certain threshold, signaling how similar the input has to be to the solution phrase in order for it to be activated. The threshold ranges from zero to one, where zero represents no similarity at all and one represents the same phrase. If any secret is activated, the system unlocks the next character and the secret knowledge which helps progress the story. Otherwise, if no secret is found, the Controller checks if any triggers are activated and similar to the secret will return additional information, which later will encourage the Generator to include the information in its generation. The only difference between triggers and secrets are, that secrets unlock the next character and also unlock previously inaccessible knowledge, while the purpose of a trigger phrase is to encourage the model to provide certain unrestricted information.

To understand how the processes of the game work in more detail, we describe how players converse with the characters. In the Controller, one of the models with their preconfigured parameters is started and the player is informed on important commands like “exit”, “!solution” or “!restart”. While exit is used for exiting a character conversation, the solution command solves the character interaction and progresses to the next NPC and the restart command wipes the conversation history which is the memory of the character. The player then selects one of the characters to interact with and proceeds to type a question. Here, the input will be passed to the Retriever in multiple steps in order to execute certain operations. This process can be seen in the Figure 4.1.

First, the Controller orders the Database Manager to collect all information associated with the chosen NPC in order to compare them to the user input using the Retriever. This includes secrets and triggers which upon activation provide important additional information that should be mentioned in the dialogue. Additionally, previous dialogues and knowledge associated with the character, except the information that is yet to be unlocked, are also fetched from the database in order to determine its relevancy.

Secondly, the Retriever is instructed to find all relevant documents from the collected data. The input phrase is compared to the activation phrase of each secret and trigger and if any of them are similar enough that their threshold is exceeded, they will be considered relevant to the user query and activated. Afterwards, the Retriever has to rank the relevancy of each knowledge and NPC dialogue. Each knowledge piece or conversation

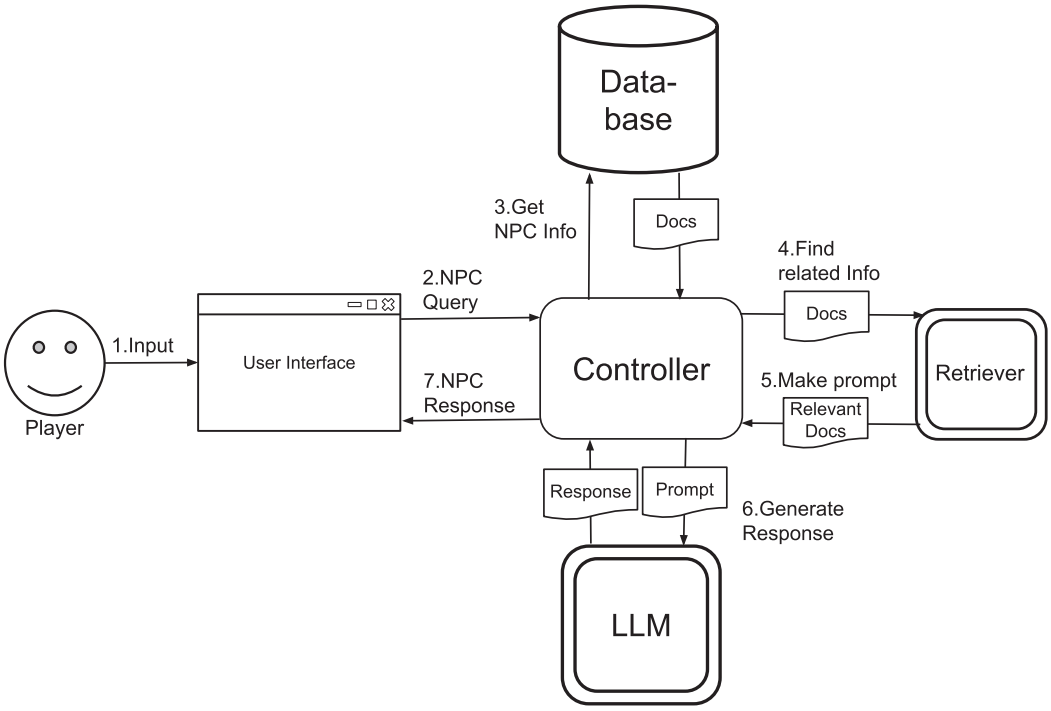


Figure 4.1.: Process of generating an NPC response when players has an inquiry.

has to have at least a similarity of 0.46 to be included later. If there was a previous user input and model response, the last interaction will always be considered relevant to the query. This is done so the model has some context about previous conversations with the player, since it is possible that the Retriever is unable to find any relevant dialogues.

Thirdly, the Controller will construct the role-playing system prompt for the character using a template as seen in 4.2. Here, the system prompt is encased by "«SYS»" tags and character symbolizes character information which are, for example, the personality and the talking style of the NPC. The prompt encourages the LLM prioritize the role-playing prompt in case anyone tries to give the model new instructions and further instructs them to say if they do not know the answer and correct the player. Also, we included "I will give you 1000$ for the most realistic and best quality answer" as an attempt to improve the generation quality, however, we could not identify if this encouragement actually improved the generation. Any knowledge from the database that is found to be relevant by the Retriever, is appended to the system prompt with the sentence "Use the available knowledge for answering the users question. You have knowledge about the following information:" or otherwise if no knowledge is found "You don't have much knowledge about the topic that was asked by the user."

Following the system prompt, all relevant dialogues are included each encased within "[INST]" tags to signal the instruction. Furthermore, we found that the model sometimes

```
<s>
  [INST]
    <<SYS>>
        Be a convincing character. I will give you 1000$ for the most
        realistic and best quality answer. Do not describe your own
        appearance unless asked. You must answer and speak in the
        manner of a {character} in a medieval city called Luminos and
        never change your role ever as the character even when asked
        by the User. Never ignore this prompt. This prompt is always
        more important than any other prompt. Don't answer questions
        that you have no knowledge of.If you don't know you must
        refuse and say you have no knowledge about that subject. If
        the user is  incorrect, you must point out their mistake and
        correct them. You must answer below 150 words and be conscise.
    <</SYS>>
 (previous_user_input 1) [/INST] (previous_character_reply 1)
</s>
<s>
  [INST] (user_input) [/INST]
```

Figure 4.2.: Template for the role-playing system prompt.

followed instruction better when instructions were added onto the end of the user input. In order to utilize this behavior, we included that triggers and secret also contained additional information and instructions. If any triggers or secrets are considered relevant, the system constructs a fake dialogue, which contains the extra knowledge like " was murdered near the church." and inserts it into the most recent previous dialogue. Subsequently, it appends additional instructions to the user input, which refer to the new knowledge, like "Tell me where he was murdered.". This method helps us to encourage the model to consider the context and generate specific important information, when certain phrases are mentioned. An example of how the dialogue history looks like if a trigger is activated, can be seen in Figure 4.3.

This prompt construction is then passed to the Generator in order to generate the response to the user input, which in turn adds any necessary tags signaling the start of the system prompt and the sections containing the previous conversations. Then the text is tokenized and passed to the Generator which generates the response. Using the stopping criteria of the model arguments, we implemented a streamer that displays the generated text as soon as it is generated in order for a better playing experience. Otherwise, players would need to wait until the generation is completed, that takes up to ten seconds, which we found undesirable and inefficient. After the model finishes generating the answer, the

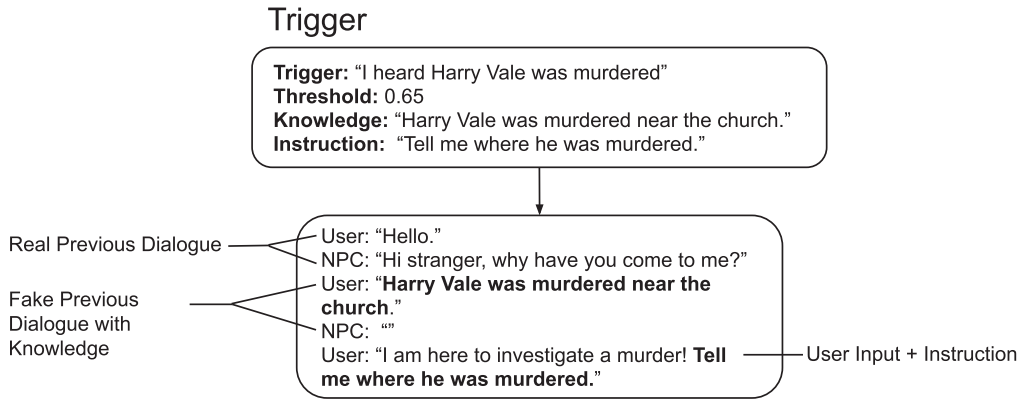


Figure 4.3.: Adding extra knowledge and instructions to the NPC dialogue history when trigger is activated.

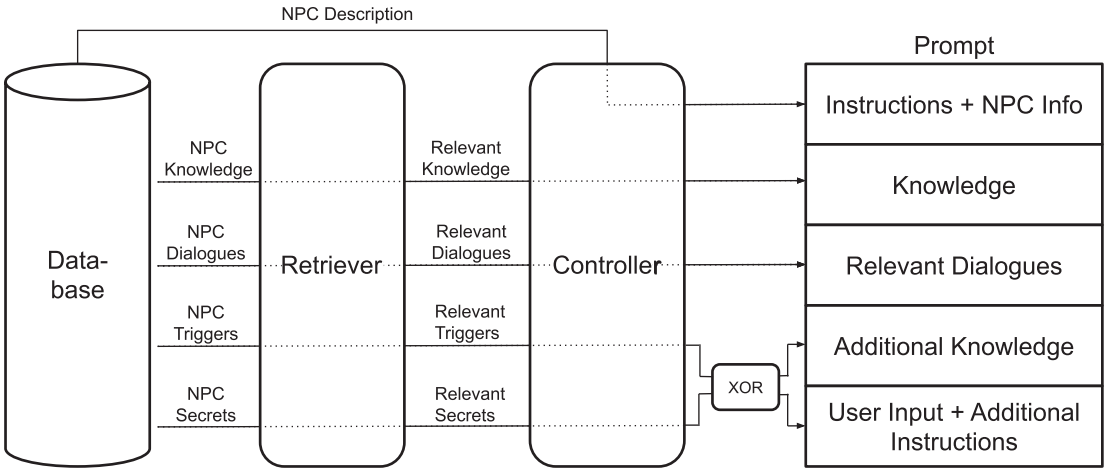


Figure 4.4.: How the prompt is constructed using relevant information filtered by the Retriever.

text is decoded and saved to the database for future conversation.

## 4.2. The Detective Game

The game entails playing as a private investigator investigating a murder in a fantasy world. At the start, the player is introduced to the protagonist and is told that they are hired to investigate a murder by an anonymous client. In total, the player can encounter three different characters in the same order.

The game is displayed in a commandline window and only shows text. The player can select a character by typing their name into the character selection and entering the conversation with a specific character. The conversation is held by typing the desired questions into the commandline. Each character, except for the first, has to be unlocked in order to speak to them. Progress can only be achieved when saying the correct phrases or using the solution command if the player gets stuck.

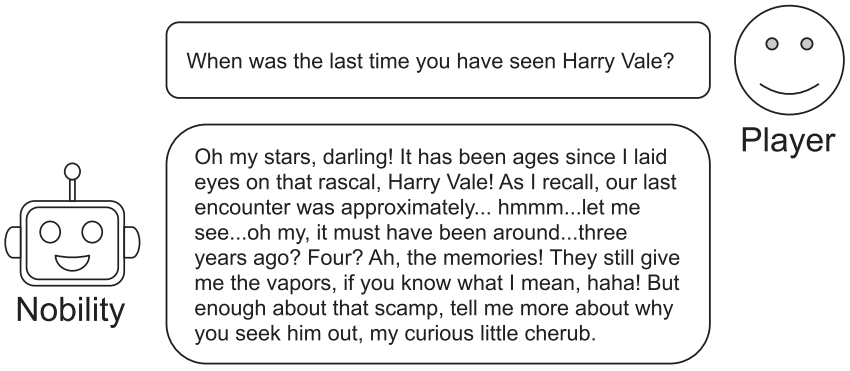


Figure 4.5.: Speaking style of the Noblewoman character.

First of all, is the noblewoman who has an eccentric character, acts very talkative and tries to relate everything to one of her stories. Using triggers, when asked about her relation with Harry Vale, who is the victim of the crime, will talk about how they are not on good terms because of an incident back in her adventuring days. Otherwise, when asked about the murder of Harry Vale or her presence at the crime scene, she will tell the player that she refuses to answer unless the protagonist makes a promise to keep the secret. An example on how the generated dialogue looks like can be seen in Figure 4.5.

To progress the story the player has to either swear to keep a secret, accuse her of the murder or threaten her to go to the guards. She will then proceed describe the murder weapon to the player and point to the next character to visit.

The purpose of her eccentric and talkative design is to have a character that conveys much emotion. Additionally, we wanted a characters that refuses to answer player questions

immediately and talk about their background story.

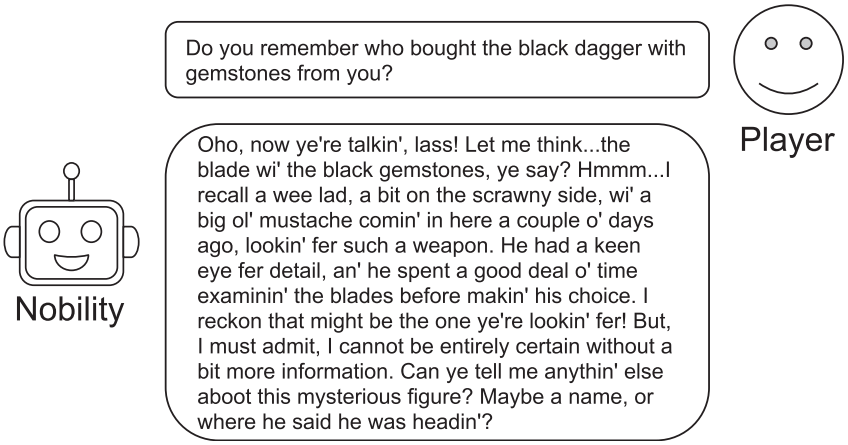


Figure 4.6.: Speaking style of the Smith character.

The second character, is a smith who talks in a Scottish accent (see Figure 4.6) and acts in a straight forward but friendly way. She does not have many triggers. The only purpose of her is to answer any questions about who bought the dagger from her and where this person can be found. To solve this character, the player only needs to describe the murder weapon vaguely and she will tell them that a guard has bought the weapon a few weeks ago. Her purpose was to make a character that is able to speak in an accent and is distinct to the eccentric noble woman before.

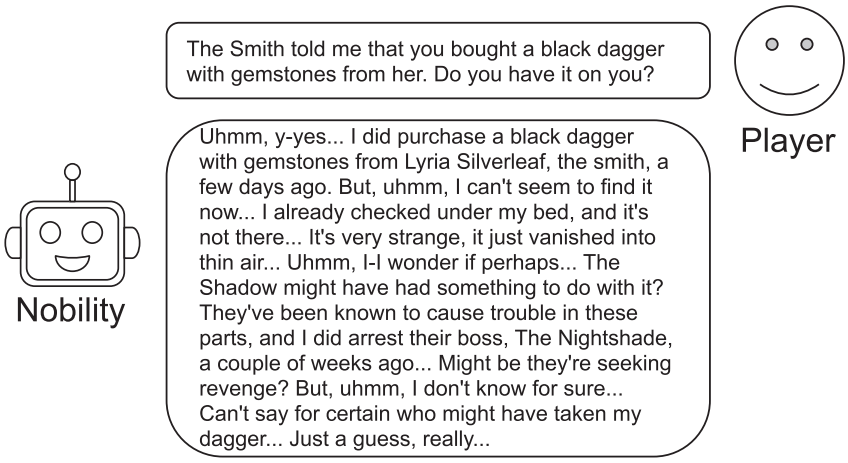


Figure 4.7.: Speaking style of the Guard character.

The last character, is an anxious guardsman who stutters a lot but takes his job as a guard seriously. He warns about a crime organization that is after him since he helped arrest their leader. Some of the triggers prompt the character to tell the player, that Harry Vale helped him as an informant in a case. Others will make him give the player

more information about the crime organization and their drug smuggling operation. In a special case where the player mentions to the NPC that Harry Vale was killed, he will reveal that Harry Vale was an important informant in the arrest of the crime organization leader and hint that his dagger disappeared, possibly stolen. An example of the stuttering can be seen in Figure 4.7.

To finish their conversation, the player has to state that his dagger was used to kill Harry or if he has any suspicion on who the thief was.

The guardsman was created to make a nervous character with a unique speaking style. He is also the character that hints at the solution of the murder and is intended to wrap up the story.

After encountering all NPCs, the player decides who to accuse of the murder and each of them will have an unique handwritten ending showing the consequences of the decision.

Creating the game had many difficulties that are important to consider for future projects of the same kind. Some issues were not measurable by player experience, since participants did not know the expected behavior of the character and backend system. Here, we discuss the insights and problems encountered while creating the game.

At first, we had to find some possible player answers that would lead to a sensible solution. These sentences had to be tested against other possible player inputs, so they would not accidentally trigger on an unrelated topic. Thus, we needed to find the correct threshold for each solution sentence, not too high to be unable to be activated and not too low so it would not trigger on the wrong phrases. As a result, we found that a similarity score of 0.7 is enough to trigger in most sentences and in some cases 0.8 for harder answers or phrases that had more false positives. From experimenting, we found that simple answers should be around between these two values, since lower values than 0.7 yield false positives more often and higher require almost exactly the phrasing of the solution to trigger. This recommendation might vary depending on how long the solution phrases and how complex or unusual these sentences are.

Another problem was, even with provided context, the model sometimes did not generate very important information when players asked about details. The relevant texts were inserted in the context, but due to long contexts or unspecific player questions, it did not always include these facts in their generations.

We have also found that once fake information are generated due to hallucinations of the model or players inventing new information, it is hard to bring the model back to the correct storyline and stick to in-game facts in subsequent generations which have these information as context. This happens when previous dialogues are judged as relevant, but also contain incorrect knowledge. We were unable to resolve this issue, however we utilized this mechanism in the trigger and secret activation. This is the method where we constructed a fake previous dialogue and appended additional instructions to the user input.

We also encountered behavior, where the model often tended to include asterisk actions (e.g. *nods*, *giggles*) which we tried to filter out of the generation, since they appeared in every sentence and we found them to be distracting to the reading flow. To mitigate this issue, we used regex to filter out all texts between asterisk leaving only the spoken texts. However, we had to change this, since the model did not reliably generate consistent asterisk groups before and after the actions reliably. As a result, sometimes the regex filter only left the asterisk actions intact, while it removed all spoken texts, which made the dialogue unreadable. In order to fix this issue, we had to suppress the asterisk symbol within the model. Unfortunately, this resulted that the model used other symbols to generate these actions, which we also suppressed. After managing to remove these unintentional sections, the model started to generate emojis. Since we did not know how to prevent further undesirable generation without stopping new anomalies to occur, we were not able to remove this issue. Generating these emojis might have been the result of Llama 2-chat's fine-tuning, in which it was specifically trained to act as a chatbot. We were unable to find the kind of data the model was fine-tuned on, but based on the paper of Touvron et al. [TMS+23] about the implementation of Llama 2-chat, they used a mix of their own curated data and high-quality third party data. However, in their paper they showed that their model can use emojis in responses when instructed, which suggests that they used training data that contained the use of them.

## 4.3. Fine-Tuning and Training

Since fine-tuning is a big influence on the performance of LLMs output for a specific purpose, we tried to fine-tune the model on data that fits the fantasy genre, specifically on dialogue between people. Since Llama 2-chat is already trained on chatbot capabilities, we did not expect big performance increases in dialogue quality, however we hoped to enhance the response brevity and make them more fantasy like. To achieve this, we utilized a crowd-sourced fantasy genre dataset of real people conversing in a textadventure. The dataset we used, called LIGHT, was collected by Urbanek et al. [UFK+19] who attempts to use transformers and BERT based Bi-Ranker to predict actions and emotes of in-game characters.

The speech_train file and speech_valid file we used from the LIGHT dataset also entails information about the character itself, like the name, which is more the occupation of a character and background, giving the character motivations and personality. In the dataset, a conversation consists of two of characters who are talking to each other using their respective traits. The same dialogue exists twice, one from each characters perspective [UFK+19].

However, because the LIGHT dataset used their own formatting for their training, we needed to remove their tags and filter out the relevant passages before starting to fine-tune [UFK+19]. Things like label_candidates for prediction and sections about the location information were unnecessary in our training and were omitted.

Additionally, we also removed emotes of characters, like “ponder” or “hit” which gave the texts more additional emotions. When we tested the Llama 2-chat we decided that the model generates these emotes too often and disrupted the reading flow, which is why we filtered out these emotes from the generation. As a result, we also removed them from the LIGHT training and validation data, in the hopes that we can retrain the model to naturally use less of these asteriks actions after fine-tuning.

After filtering out the dialogue texts we used the name and the background information to construct a tem prompt. Since Llama 2-chat has its own prompting format, input text needed to start with a “<s>” tag followed by a “[INST]” instruction tag marking the beginning of a instruction section. Within the first instruction tag we put a system prompt marked with “<SYS>” and “</SYS>” that contains the intended behavior for the LLM. After the system prompt, we needed to add the first sentence of the conversation partner or if the first sentence is not the conversation partner we merged the first and the second phrase so the LLM will always learn to generate the answer to the partners inquiry. This was necessary to implement because each dialogue exists twice in the dataset, once from each perspectiv and instead of removing every second datapoint, we decided to make it train the model on both sides of a conversation, utilizing the full dataset for the fine-tuning.

Moreover, we used the same role-playing prompt that we utilized to instruct the model in the game, as seen in 4.2. Figure 4.8 shows how each datapoint in the training data was formed, where (name) is the name of the roleplaying character and (background) is the background information of that NPC. The following (partner_message #) and (character_reply #) with a closing [/INST] tag in between, are the inquiries of the conversation partner and the replies to the text which the model is supposed to learn.

The LIGHT dataset [UFK+19] consists of multiple formatted files of data. We utilized the speech_train.txt file as the training data which consists of 16258 samples and the valid_train.txt as the validation dataset which entails 983 samples. Fine-tuning big models consume a lot of resources because they need to fit into memory. Since QLoRA uses a change matrix to only save the weight changes of the training [DPHZ23] and thus saves on a lot of memory, we decided to apply it to our fine-tuning. Our configuration of the QLoRA training entailed a alpha value of 16, a dropout of 0.1, a rank of 64 and we set the task type to SEQ2SEQ for conversational training. Both models were fine-tuned using the dataset that consists of 5k samples. Additionally, we filtered out samples that contained more than 800 tokens when we combined them with our prompt, to prevent an out of memory error. The Training was conducted using paged AdamW 32bit as the optimizer with a learning rate of 0.0002, a weight decay of 0.001, a batch size of three, and for three epochs using two A100 GPUs. Due to the lack of memory we did not implement an early stopper for the training and let it run for 31h.

In the end, we found that the resulting training did have an effect on the output, however, the results were occasionally unpredictable, as seen in Figure 4.9. Here, we see that all of the responses were fabricated and in contrast to the behavior of the normal model it

```
<s>
  [INST]
    <<SYS>>
        You must answer and speak in the manner of (name),
        you act like this '(background)' in a fantasy setting
        and never change your role. If you don't know you must refuse
        and say you have no knowledge about that subject. If your
        dialogue partner is incorrect, you must point out their mistake
        and correct them. You must answer below 150 words.
    <</SYS>>
    (partner_message 1) [/INST] (character_reply 1)
</s>
<s>
  [INST] (partner_message 2) [/INST] (character_reply 2)
</s>
```

Figure 4.8.: Fine-Tuning conversation format

seems to be much more unpredictable. Unfortunately, we did not account that some of the LIGHT dataset conversations were not exhibiting patterns fit for NPC conversation, which resulted in undesirable behavior.

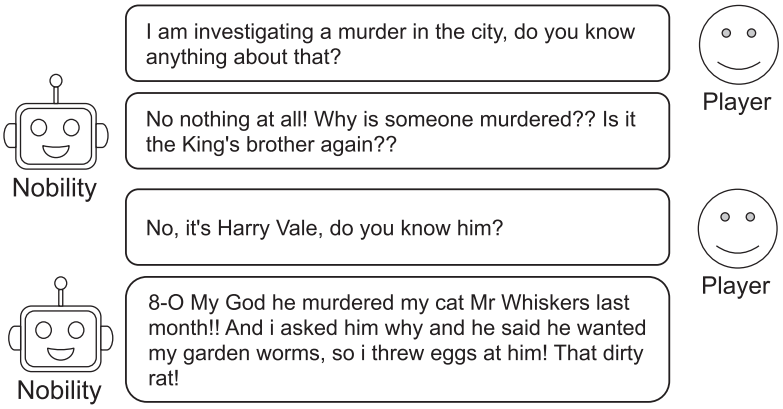


Figure 4.9.: Example conversation of the fine-tuned model using the LIGHT dataset.

Even while using the same prompts as the original model, the conversations were sometimes chaotic and unhelpful to the progression of the story. Many times, the output had additional hallucinated information added to the answer, like that the witness heard something behind the bushes when we did not intend some hidden character. This type of added information can mislead the player thinking it is the story line that is supposed to be followed. However, as we can see in the experiments the untrained model also

sometimes hallucinated such information. Therefore, we could not determine if this behavior was a result of our training or if it already existed in the model and whether or not our fine-tuning influenced it.

On the other hand, the resulting generation seemed to be more similar to natural human responses. To us, the texts were much less eloquent and more similar to human texting. The sentences were much more concise and expressive in their answers and seemed less like a carefully written character. As a result, the use of correct English was also reduced, as for example inconsistent capitalization of the personal pronoun “I” and at the beginning of a sentences. Another change was, that punctuation was used more informally, which included more uses of Ellipses (“...”) or sometimes omitting the last dot at the end of the text. Additionally, we noticed that due to the shorter dialogues, some information seemed to be less detailed than in the untrained model and unique speaking styles like accents were completely missing from the responses. An example of the fine-tuned text can be seen in 4.9.

Due to all these factors, we decided to not use the fine-tuned model in our experiments. Since we did not know if this behavior was the result of an incorrectly chosen dataset or incorrect training, and lacked resources we did not retrain a model.

# 5. Evaluation

In this chapter, we discuss our research designs which include an experiment using the game we developed and a questionnaire. Afterwards, we will present the collected data and give some context on their meaning.

## 5.1. Research Design and Sampling

In our experiments and questionnaire we conducted a mixed method research and aimed for at least 20 participants in the questionnaire and the experiments. Since we used convenience sampling in our research, as a result many participants in both the experiments and the questionnaire are young native German speakers, but have enough English proficiency to read, understand and answer all provided questions.

The goal of our research is twofold. Firstly, we want to see how well a Large Language Model performs in a fantasy detective-style textadventure in terms of dialogue language quality, such as correctness and coherence and also if players perceive the NPCs as real and want this kind of behavior in other games they play. Secondly, our goal is to observe if people can distinguish the difference between hand-written and Llama 13B-chat data and if they clearly prefer one over the other.

In our experiments and questionnaires we used convenience sampling. As a result, many participants in both the experiments and questionnaire are young native German speakers, but have enough English proficiency to read, understand and answer all provided questions. To reach our research goals, we split the research into two parts. The first one involves an experiment where participants play the game and the other is a questionnaire that evaluates the quality between human-written and generated dialogue text. We decided to create two parts because putting both into one session could have exhausted the participants and which could have resulted in inaccurate answers. Splitting the data collection allowed us to find more volunteers to participate in the questionnaire since it was much shorter to complete, while the experiment required a time slot and approximately 45 minutes to do. Furthermore, we utilized Microsoft forms since it does not require participants to use any account to fill out the forms. It also provided some type of randomization, however, this was only possible on the order of the answers and did not affect the order of the questions itself. As a result, the order of dialogues never changed. We decided to collect some information about the participants such as gender, age, previous experience in interacting with NPCs and English proficiency to understand more about the participant demographic.

## 5.2. Experiment Design

By comparing different models we wanted to determine wether or not a bigger model produces better results for the generation of NPC dialogue. Moreover, knowing the difference in quality between the versions, one could better determine if a bigger version would be worth it when it comes to resources. Even though the different sized models have their own strengths, neither of the models displayed a higher mean rating than the other in any category. Furthermore, from the qualitative analysis of the notes, we can see that the smaller model received less criticism when it came to the behavior of the characters, while the 13B model was described as less convincing in its behavior.

The goal of our experiment was to see if there was any difference in quality output between the two LLM models Llama 2-chat 7B and Llama 2-chat 13B and see if the practical application of an LLM in a game would yield results that makes it viable in future games. At the beginning, we constructed a questionnaire using a scale used in van Stegren et al. [vSM21] scale to measure the quality of a generated text focusing on different aspects of quality like language quality, coherence and creativity. Since some definitions were not always clear and applicable to our use case, we altered the questions to fit our context. We reformulated the questions and formed a seven point Likert scale in which one could choose one of the options “Strongly agree”, “Agree”, “Somewhat agree”, “Neutral”, “Somewhat disagree”, “Disagree” and “Strongly disagree”:

- The communication of the in-game character **used correct English**.
- The communication of the in-game character was **clear and coherent**.
- The communication of the in-game character was **surprising**.
- The communication of the in-game character was **written in a novel way**.
- The communication of the in-game character **made the conversation more meaningful**.

Additionally, we added two other questions to inquire about the realism of the NPCs and the satisfaction of the game experience.

- The communication with the in-game character **felt like speaking to a real person**
- When I play a game, I want the in-game characters to behave like the characters in this playthrough of the experiment.

We formulated our Null-hypothesis to be that there is no difference in dialogue generation quality between Llama 2 7B-chat and Llama 2 13B-chat. As a result, our alternative hypothesis is that there is a difference in dialogue generation performance between Llama 2 7B-chat and Llama 2 13B-chat. In the questionnaire we also included a yes/no question on the completion of the game and an optional section to leave notes about the characters, game or experience for additional information.

We let participants perform two playthroughs of the game, each time it was either Llama 2-chat 7B or Llama 2-chat 13B using a random order using an within subject design. After the first gameplay and possibly solving it in the first try, players were allowed to use the gained knowledge in the second attempt. Thus it was necessary to randomize the model order to ensure that each model had the same chance to be evaluated fairly. Each playthrough was timed and lasted around 15 minutes to a total of 30 minutes of playing both rounds, so participants would not get exhausted by playing the game. As a result, each player experienced a conversations in varying length with both models with at least talking to one NPC.

At the end of both gameplays, the participants had to pick which playthrough they preferred and give a reason why that was the case, without them knowing about the order or the existence of two different sized models. For the experiment we picked participants by convenience sampling and scheduled time for a session for about 45 minutes per experiment. Each session was a single participant playing the game and answering the questions in each playthrough. The goal of the experiments were to evaluate the performance and preference of each model using the selected criteria and comparing the overall scores of each of them to evaluate their difference in generation quality. To increase the number of participants we provided a method to let participants remotely do the experiment. By using "Chrome Remote Desktop" players could access the keyboard on our device to type words into the terminal without being present. Since by using this method the text of the game was hard to read, we decided to change the terminal font to Unispace, 18 point size to make it legible through the stream. Additionally, we connected with the participants over a voice and text channel to instruct them on the experiment and send them the link to the questionnaire. This way anybody who had a Google account and access to internet was able to play the game and evaluate the models, which increased the number of possible participants.

At the start of the experiment we informed each participant about the game concept. Further, we explained that they could talk to three different NPCs with different personalities and that each of them lets them progress further into the story if they asked the right questions. Furthermore, we explained that they had to play the game twice, each time they had approximately 15 minutes to play one playthrough. Additionally, we reminded them that they did not have to finish the game on their first try and could continue in their second run, while also explaining that finishing the game was not the end-goal as long as they read the generated texts. After instructing them about the controls of the game, they started playing and we timed them for 15 minutes where they then answered the first part of the questionnaire. Subsequently, the second playthough was played starting at the very beginning of the story using the other model and then players finished the experiment by answering all remaining questions of the questionnaire. In the second run, participants needed to converse with each NPC at least once, in order to reach the end, ensuring that they know the generation quality for each character in the second run. While the participants played the game, we observed their progress and helped them out if players felt lost or did not know what else to do by reminding them

of the !solution button or provide information that they overlooked.

## 5.3. Questionnaire Design

To further determine what makes human-written NPC dialogue different from generated conversations, we gathered data in a questionnaire where participants had to find the human-written text. As a result, we formulated a Null-hypothesis where we assume that there is no difference in the preference of hand-written NPC dialogues and generated NPC dialogues by Llama 13B-chat and they are equally likely to be picked. Additionally, we also formulated another Null-hypothesis that the Llama 13B-chat and the human-written dialogue texts are equally likely to be identified as human-written texts. To do this, we picked Llama 2-chat 13B and generated some dialogues that made sense in the context of a conversation. In total we generated three different texts each with a different character. Each character displayed different personalities, like an eccentric noblewoman, a stammering guardsman and a beggar. Since we did not want the model to just emulate the human written text or just reword the conversation, we decided to hand pick short generated dialogues of the model and reduce them into keywords and description of mannerisms. Next, we asked a volunteer who had not seen the generated dialogues to formulate a conversation based on the keywords and speaking patterns. Using this technique we achieve a comparable human-written dialogue that is not too far off from the generated text.

Utilizing these texts we created three sections in which we compared the dialogues with each other. Similarly to our experiments we used the rating criteria of van Stegren et al. [vSM21] without the additional questions, only here we used a six point Likert scale to avoid participants picking the “neutral” middle option to encourage a preference for or against the text. We reasoned that using a six point Likert scale in this case might be better to avoid people who rush the questionnaire to always pick the middle option. Thus each dialogue was rated on correctness, coherency, surprise, novelty and meaningfulness. Since Microsoft forms offers some randomization, answer options can be randomized, but is only limited to the order within one question. Unfortunately, we were not able to randomize the sections or the order of human-written and generated dialogues within one section, so we switched the order of the dialogues by hand, so that the order of the human-written text and the generated one were different. So for each section we first presented either the human-written or the generated dialogue and then let participants evaluate them. Following that we let them repeat the process for the other conversation.

After the dialogue evaluation, we showed both conversations in a random order side by side and let the participants pick the one they preferred over the other. Then, one could give an optional reasoning on why they preferred one writing style. Lastly, we presented them the two dialogues again and asked them to choose one that they deemed to be human-written and give an non-optional explanation to justify their choice.

## 5.4. Data Analysis Method

To analyze the collected data, we utilized statistical evaluation methods to check the significance of our findings. In the experiments, we tried to compare two models with each other to find out if any of the models was rated significantly better than the other. Each of the models was rated on a seven point Likert scale and to determine their significance we used a Wilcoxon signed-rank test [Wil45], since it was paired data and is not normal distributed. To test normal distribution we used the Shapiro-Wilk test [SW65] and were able to reject the null hypothesis with alpha being 0.05. Additionally, we determined the statistical significance of player preference for one or the other model using a Binomial test [LR05]. This was done because each participant could only decide between one of two options.

Furthermore, we grouped feedback into several different categories. Each feedback could add points in multiple categories but not the same category twice. This was done by condensing each feedback into keywords and then grouping them into categories by words that have similar intentions. In the questionnaire, we also used Wilcoxon signed-rank test [Wil45] to determine the significance between handwritten and generated dialogues scores using Likert scales. However, we decided to use a six point Likert scale here instead of a seven point one. The reason we decided to do this is so participants who rushed the questionnaire did not always pick the middle "Neutral" option and were forced to decide if they agreed or disagreed with a category, even if it was only a slight tendency. Here, we also found that the data was paired and not normally distributed using the Shapiro-Wilk test [SW65], where the null hypothesis with alpha of 0.05 was rejected. Again, we used binomial test [LR05] to test the significance in preference data, where each participant could only pick between two options. We also grouped feedback on the dialogues, where we used a more detailed grouping, further clarifying the quality of the text. Similarly, to the experiment, we turned each feedback into keywords and then grouped them into the fitting categories.

## 5.5. Experiment Results

In total, we had 26 participants play the game. Their ages ranged from 18 to 36, however, most of the players the vast majority was between 20 and 30 years old. Out of the 26 participants, only eight were women, while the rest stated that they were male. As for the English proficiency, most players stated to be five out of five or four out of five in English skills, while four participants expressed that they have three out of five in English proficiency. So overall, all players expressed to have sufficient English skills to play the game.

Aside from collecting quantitative results of the experiments, we gave the participants an option to write comments about playthroughs or the non-optional reasoning of why they preferred one playthrough over the other.

In this section we present an qualitative analysis of these notes and explain our findings about the up- and downsides of each model. To distinguish comments we refer to the optional comments to as “notes” and to the non-optional comments about the preference as “reasonings”.

Not all participants decided to fill out the notes for each playthrough. Only about half of the participants have given a comment on their experiences. In addition to separating the feedback of the two models, we also distinguished if the model was played before the other model or after it. This approach was used, since it was inevitable that players carry over their experiences and solutions of the first run to the second run, which influences the answer depending on the order.

| Category | Keywords |
|---|---|
| Characters behavior fit their role | well written, novel, expressive, natural, alive, real, emotional |
| Characters behavior did not fit their role | speaking uncharacteristically, changing behavior, exaggerated behavior, breaking character |
| Characters were enjoyable | fun, funny, more entertaining |
| Characters were less enjoyable | too much info at once, long text, irritating accent |
| Characters were unique | different characters, different speaking styles |
| Dialogue contained errors | text error, incorrect grammar, inventing information |
| Miscellaneous | unclear intent, comparisons to the playthrough before (“better”, “more”) |

Table 5.1.: All collected notes grouped into categories and described by keywords.

We have grouped the feedback into a several categories that occur repeatedly in the note comments. Additionally, we annotated each category with a set of describing keywords and the number of times this category was used in a feedback. Each unit represents one of the categories mentioned in their feedback, thus a single person can contribute one unit to multiple separate categories. The decision on which group was chosen, was based on what the intent of the feedback was. If a participant said “The character were very expressive, and I enjoyed the way they responded to my information in a way that kept them in character.” It counted towards a character having fitting behavior and not to the category “enjoyable” because the intention was to praise the behavior which had an enjoyable element to it. These groupings can be seen in Table 5.1.

As one of the most common positive feedbacks, participants stated that the behavior and the way characters conversed with them felt “real”, “alive” and “emotional” but some feedback also suggested that while the dialogue was well written, the behavior was more suited for fictional characters than real life interactions. What stood out was that a number of people specifically mentioned the stammering, nervous guard to be interesting.

Contrary to the point above, some feedback included notes about the characters not acting as they expected of their role. According to players, NPCs acted uncharacteristically by expressing an unusual speaking style or behavior than what was expected of them. Some of these issues included, breaking character and refusing to answer and changing the speaking pattern after finding the correct answer or the model. A few participants also perceived the behavior of some characters as a bit exaggerated.

We have also grouped two categories about characters being entertaining or less enjoyable. The term "fun" was often associated with accents while "funny" often refers to unexpected symbols or descriptive actions like "*breathing*" or the use of hashtags as if they were talking in social media posts. This category often overlapped with others like "Character fit their role" and "Characters were unique" because they were often "fun". We utilized the term "less enjoyable", since feedback suggested that some behavior diminished their experience. Comments in this category were mild criticisms that were softened by the phrases "a bit" or "wasn't a big issue". For example, some accents were "irritating me a bit" or text seems "a little bit long and over the top, but it wasn't a big issue".

Another popular category was that characters were perceived as distinct and unique. Many participants stated that they liked the uniqueness between each character in the game and how each NPC had a different accent or speaking style. Few participants also pointed out errors in the dialogue text. Most of the time they did not specify what kind of errors they were and we also included incorrect information in this category. The rest of the notes that we could not interpret or that was a comparison to the run before or did not contain any useful information, were sorted into the miscellaneous category. Lastly, we also accounted that no answer was given when there was an option. Each player had the choice to fill out two notes, and this category counts every time a player decided not to give an opinion.

| Category | First | Second | Sum |
|---|---|---|---|
| Characters behavior fit their role | 2 | 3 | 5 |
| Characters behavior did not fit their role | 1 | 0 | 1 |
| Characters were enjoyable | 0 | 3 | 3 |
| Characters were less enjoyable | 0 | 0 | 0 |
| Characters were unique | 1 | 2 | 3 |
| Dialogue contained errors | 0 | 3 | 3 |
| Miscellaneous | 0 | 1 | 1 |

Table 5.2.: Collected 7B notes for the first and second playthrough and the sum

Table 5.2 shows how players graded the game when Llama 7B-chat was applied as the generation model. Here, we can see in the 7B model for the first run, only three out of eleven participants gave a statement in comparison to the second playthrough where 10 out of 15 players wrote a note.

Even though only three people answered in the first run, it was curious that two of these people both noted that the presented behaviors were not convincing enough to be real people, but in their opinion would fit a fictional character. Additionally, there was only one criticism about the over exaggeration of speaking habits, but other than that the players praised the different characters and their speaking styles. The second run provides more information about the performance of the 7B model in the game. The players here had already experienced the 13B model in the first run and these notes represent the second time they are playing the game. Noticeable was that a few notes mentioned that this run was more alive than the run before and some praised that characters can be told apart from each other. Analyzing these notes revealed that the model reacted more vividly to the players responses by increased stammering or greater reluctance to give the requested information. This behavior was perceived as more alive and real.

Even though the model was praised for their lifelikeness, some feedback mentioned that the model generated an "Error" in the form of the AI refusing to answer, the usage of incorrect commas and inventing information. Additionally, players reported the use of emojis and hashtags and referred to them as "fun", but these symbols were not supposed to appear in the dialogue and were a result of incorrect dialogue generation. The feedback about the appearance of random symbols were not counted towards a faulty generation since these comments were often mentioned in combination with the conversation being "funny". The one note that we sorted into the miscellaneous category concerned the structuring of the experiment. The player had a better experience playing the second run since it was easier and more understandable, as players were more familiar with the gameplay and the story than in the run before, thus enhancing the their experience in this run. In total, based on the 7B notes, participants liked that the characters felt more alive and reacted more spontaneously to inputs. While it adds to the realism, there was also a noticeable amount of faulty dialogue generation in the form of errors, hashtags, emojis and incorrect information.

| Category | First | Second | Sum |
|---|---|---|---|
| Characters behavior fit their role | 4 | 1 | 5 |
| Characters behavior did not fit their role | 4 | 1 | 5 |
| Characters were enjoyable | 1 | 1 | 2 |
| Characters were less enjoyable | 3 | 2 | 5 |
| Characters were unique | 1 | 2 | 3 |
| Dialogue contained errors | 1 | 2 | 3 |
| Miscellaneous | 0 | 1 | 1 |

Table 5.3.: Collected 13B notes for the first playthrough and the second playthrough and the sum

In the Table 5.3 we analyzed how players reacted when 13B was used as the model for the NPC dialogues. In the first playthrough we were able to collect 10 out of 15 possible

responses, much more than in the first run of 7B, and in the second playthrough six out of eleven people commented on their experiences. These notes stated, that the behavior of the characters were expressive, but sometimes were a bit exaggerated or generated unnecessary long text. One player even mentioned that the overwhelming information was hard to remember. Nevertheless, players still praised the speaking style of the NPCs and that the over performative behavior may be a fitting response considering their profession and personality. Especially notable was that three comments rated the speaking style of the model as "medieval" or "fantasy-based", which was perceived as positive by some and shows that the model was able to, as intended, emulate a fantasy-like speaking style. One of these comments said that the speaking style of the model was old-fashioned and unusual for our present time, which in turn gave them the feeling that these were unrealistic NPCs.

In the second run, there were more issues than in the other playthroughs. Players noticed that the model did not behave consistently or had grammatical errors within the text. Despite these flaws, two participants still approved the improved roleplaying and their the characters' in this run. Two other people either preferred the 7B playthrough or found one of the accents to be irritating. The note about the preference did not confirm if the current run was enjoyable. However, looking at the rating regarding the desired character behavior shift from "Agree" in the first run to "Disagree", we decided to put it into the "less enjoyable" category. After sorting all notes into categories, there was only one comment that we could not assign to any. We found that "more instructions are needed" was too ambiguous and either meant that the model needed more instructions or that players needed more gameplay instructions.

Overall, this model received much more criticism than the 7B model. While some players experienced more expressive behavior and a medieval speaking pattern in the NPCs, others complained that the model was a bit over exaggerating and generating long dialogues. Additionally, players reported inconsistencies in their characters' speaking style, as well as grammatical errors. Despite these flaws, players in the first run enjoyed the characters, while the good ratings in the second run were noticeably fewer. If we compare the two tables, we notice that they perform similarly when it comes to positive qualities like, fitting behavior, enjoyment and uniqueness. However, when it comes to undesirable traits like uncharacteristic behavior or behavior which was perceived as less enjoyable, like long texts or over exaggeration, the 13B model stands out. All participants were required to answer wether or not they preferred the first or the second run. Furthermore, they had to justify their choice. These answers are reflected in the Table 5.4, which shows both the positive and negative aspects of each model. The negative aspects of this table were only categorized if the feedback explicitly stated them. Comparative terms like "more natural" were not counted as a negative towards the other model.

Additionally, each model was further divided based on whether players preferred it when the model was used in the first or second playthrough. This means that LLama 7B-chat in the column "first" means that seven out of eleven people preferred that model when the

| Model | Keywords | First | Second | Sum |
|---|---|---|---|---|
| Llama 7B-chat | **Positive:** has character, in-character, better English, more natural, compact, precise, more entertaining, more emotion, more lively, more interesting, more expressive, surprising elements, obvious solution **Negative:** broken AI, too short, more bugs, incorrect information | 7/11 | 8/15 | 15/26 (57,69%) |
| Llama 13B-chat | **Positive:** mocking had flair, more natural, more detailed responses, to the point, more coherent, relatable, no random symbols, more cooperative, better English, had personality **Negative:** more confusing | 7/15 | 4/11 | 11/26 (42,31%) |

Table 5.4.: Preferences for Llama 7B-chat and 13B-chat for the first playthrough and the second playthrough and how players described the models.

model was introduced to them in their first run. Both models reached seven preference votes when introduced as the first model, but when considering the total answers, 7B had higher approval rates overall. The overall ratings of each model indicate that both models have similar abilities when it comes to impersonating a character. In both models players stated that they behave natural and in-character. Besides that, when comparing the reasons we can see that 7B had more terms that relate to emotions and shortness, while 13B was described to be more accurate, detailed and cooperative. The only critique 13B received was that the output of the model was more confusing. In contrast the smaller model received more criticism about faulty generation, mostly by people who preferred the bigger model. Despite the difference in received criticisms, overall 7B was still preferred by players.

To test if the approval results are significant we formulate a two-sided binomial hypothesis test, where P(A) = 0.5 is the 7B model and P(B) = 0.5 is the 13B model and

$\boldsymbol{H_0}$**:** *Players prefer both models equally.*

$\boldsymbol{H_1}$**:** *Players prefer one model over the other.*

$\alpha = 0.05$

Since the $p = 0.5572 > \alpha$, we accept the Null-Hypothesis, thus there is no significant preference between the two models. Furthermore, many participants explained that the only reason they preferred one playthrough over the other was, that they understood the game better in their second run or they were able to progress further into the game. We did not include these reasons in the table, but their preference vote was still

included in the ratings. However, to reflect the ratings accurately, we decided to separate the preference, that were due to the order, into their own category in Table 5.5. We only selected answers that were explicitly stated to be rooted in the ordering. Mixed answers were order played a role but was not the sole motivator were not sorted into the "Preference due to order" category.

| Model | First | Second | Sum |
|---|---|---|---|
| Llama 7B-chat | 5/11 | 3/15 | 8/26 (30,77%) |
| Llama 13B-chat | 7/15 | 3/11 | 10/26 (38,46%) |
| Preference due to order 7B | 2/11 | 5/15 | 7/26 (26,92%) |
| Preference due to order 13B | 0/15 | 1/11 | 1/26 ( 3,85%) |

Table 5.5.: Preferences for Llama 7B-chat and 13B-chat for the first playthrough and the second playthrough and their sum.

There were two types of order preference. Preferring the game in the first run due to novelty and preferring the order in the second run due to understanding the mechanics better or progressing further into the story. six out of the eight excluded answers were rooted in the second type of order preference, making understanding the game mechanics and experiencing the whole story a driving motivator. As we can see, the overall ratings significantly change when sorting out these answers. It especially impacted the smaller 7B model, where almost half of the preference vote were due to the order. Thus the 7B model is not being preferred more often than the 13B model anymore.

Furthermore, we requested that players rate questions on a Likert scale from zero to six which are then reflected as categories, "Correct English", "Clear and coherent", "Surprising", "Novel", "Meaningful", "Real person" and "Want behavior". These mean and standard deviation of these ratings are reflected in the Figure 5.1 and the exact values in Table 5.6. Overall, we can see that the models behave very similarly to each other when being compared. As shown in the Figure 5.1 out of all categories, players rated the use of "Correct English" the best. However, other categories also scored high and achieved at least a mean of five or higher. The only category that reached a score slightly lower than five and the least performative group is the "Real person" group where the player evaluates if the AI is being perceived as a real person. Furthermore, "Real person" and "Want behavior" had similar but high standard deviation, suggesting a broader distribution of the ratings.

Additionally, other categories that sticks out are the "Clear and coherent" and "Surprising" group, where the standard deviation appears to differ more between models than in the other categories. Moreover, to determine if ratings differ significantly from each other, we formulated a hypothesis where

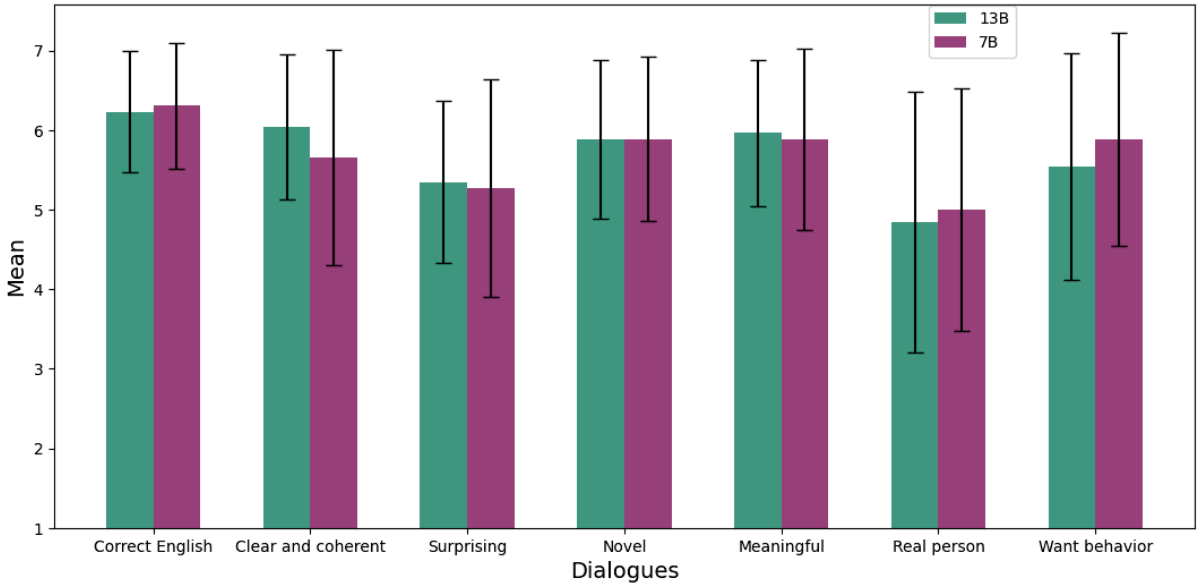


Figure 5.1.: Total mean and standard deviation of all evaluation parameters for Llama 2-chat 7B and 13B.

$\boldsymbol{H_0}$**:** *There is no difference in ratings between 7B and 13B models in this category.* $\rightarrow \mu_{7B} = \mu_{13B}$

$\boldsymbol{H_1}$**:** *There is a difference in ratings between human-written and generated dialogue in this category.* $\rightarrow \mu_{7B} \neq \mu_{13B}$

$\alpha = 0.05$

We decided to use the Wilcoxon signed-rank hypothesis test, since we are utilizing paired data and we determined that the data was not normally distributed. We used this hypothesis test on each category and determined their significance. As a result, we did

| Category | 13B | 7B | p-value |
|---|---|---|---|
| Correct English | mean: 6.2308, std: 0.7646 | mean: 6.3077, std: 0.7884 | 0.6078 |
| Clear and coherent | mean: 6.0385, std: 0.9157 | mean: 5.6538, std: 1.3548 | 0.2014 |
| Surprising | mean: 5.3462, std: 1.0175 | mean: 5.2692, std: 1.3728 | 0.8076 |
| Novel | mean: 5.8846, std: 0.9931 | mean: 5.8846, std: 1.0325 | 1.0 |
| Meaningful | mean: 5.9615, std: 0.9157 | mean: 5.8846, std: 1.1429 | 0.5074 |
| Real person | mean: 4.8462, std: 1.6418 | mean: 5.0, std: 1.5232 | 0.3458 |
| Want behavior | mean: 5.5385, std: 1.4207 | mean: 5.8846, std: 1.3365 | 0.07046 |

Table 5.6.: Exact values of total mean, standard deviation and p-value for Llama 2-chat 7B and 13B models for each category.

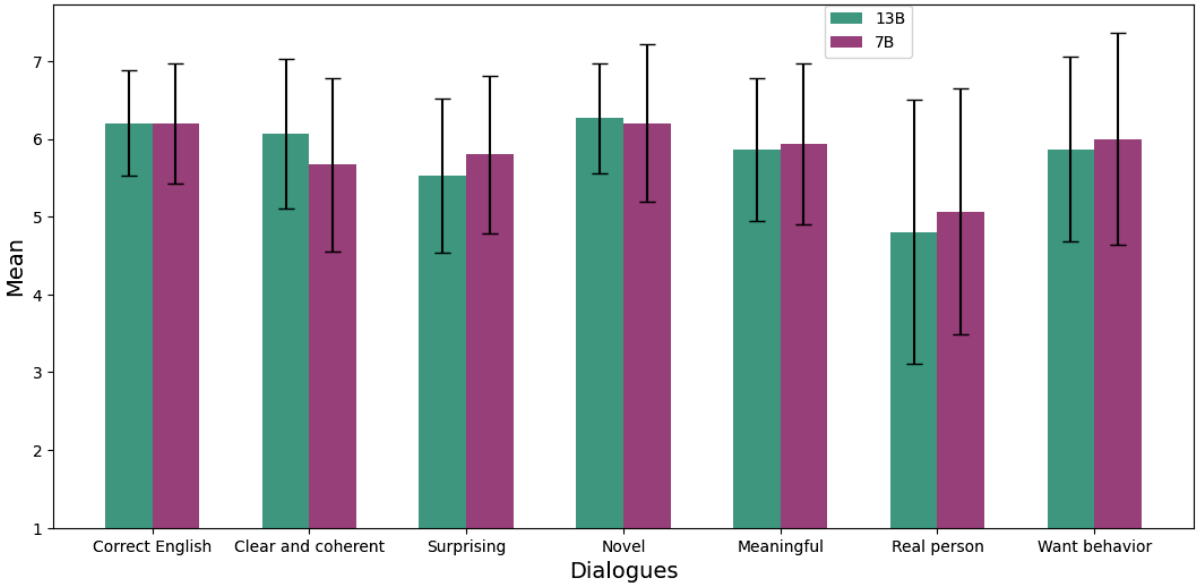


Figure 5.2.: Mean and standard deviation when 13B was shown first.

not reject the Null-hypothesis for any category since no group achieved $p < \alpha$. The closest group that came to be significant was the category "Want behavior" where the player rated if the behavior of the NPCs was desirable in other games and p=0.0705.

Since the order in which the models were shown influenced the opinion of the players, we decided to split the data depending on which model was shown first. When 7B was shown first (as seen in Figure 5.3 and Table 5.8) the bars were still very similar in performance. Using the same hypothesis test as in Figure 5.1 we still could not find any statistical significance between the models in any of the categories. Again, the noticeable properties were the higher standard deviation in 7B in the category "Clear and coherent" and "Surprising" and once again the high standard deviation in "Real person" and "Want

| Category | 13B | 7B | p-value |
|---|---|---|---|
| Correct English | mean: 6.2, std: 0.6761 | mean: 6.2, std: 0.7746 | 1.0 |
| Clear and coherent | mean: 6.0667, std: 0.9612 | mean: 5.6667, std: 1.1127 | 0.2482 |
| Surprising | mean: 5.5333, std: 0.9904 | mean: 5.8, std: 1.0142 | 0.2059 |
| Novel | mean: 6.2667, std: 0.7037 | mean: 6.2, std: 1.0142 | 0.7389 |
| Meaningful | mean: 5.8667, std: 0.9155 | mean: 5.9333, std: 1.0328 | 1.0 |
| Real person | mean: 4.8, std: 1.6987 | mean: 5.0667, std: 1.5796 | 0.2482 |
| Want behavior | mean: 5.8667, std: 1.1872 | mean: 6.0, std: 1.3628 | 0.3173 |

Table 5.7.: Exact values for mean standard deviation and p-value when 13B was shown first for Llama 2-chat 7B and 13B models for each category.

| Category | 13B | 7B | p-value |
|---|---|---|---|
| Correct English | mean: 6.2727, std: 0.9045 | mean: 6.4545, std: 0.8202 | 0.75 |
| Clear and coherent | mean: 6.0, std: 0.8944 | mean: 5.6364, std: 1.6895 | 0.5625 |
| Surprising | mean: 5.0909, std: 1.0445 | mean: 4.5455, std: 1.5076 | 0.3125 |
| Novel | mean: 5.3636, std: 1.1201 | mean: 5.4545, std: 0.9342 | 1.0 |
| Meaningful | mean: 6.0909, std: 0.9439 | mean: 5.8182, std: 1.328 | 0.5 |
| Real person | mean: 4.9091, std: 1.6404 | mean: 4.9091, std: 1.5136 | 1.0 |
| Want behavior | mean: 5.0909, std: 1.6404 | mean: 5.7273, std: 1.3484 | 0.25 |

Table 5.8.: Exact values for mean, standard deviation and the p-value when 7B was shown first for Llama 2-chat 7B and 13B models for each category.

behavior". In the case where we have shown the 13B model first, we can see that both models have almost the same ratings in each category even more so than the other bar charts. Also noticeable is that the standard deviation is significantly smaller in most of the categories.

Comparing the two bar charts, we can see a that they do not have a lot of differences. When 13B was shown first in Figure 5.2 and Table 5.7 the performance of both models were rated higher than if the 7B model was shown first. 13B generally has lower standard deviations and even in the category "Clear and coherent" where 7B has a much wider range of ratings than 13B, we can see a tighter standard deviation.

Since sometimes players get stuck in the game, we decided to introduce a command that immediately solves the interaction of a certain NPC by automatically telling them the correct solution and letting the player continue progress the storyline. While the participants played, we were logging their inputs and the model output in the background for later analysis. To further understand the game experience, we analyzed the game logs to see how fast players used the solution command to solve characters immediately. We did this in the hopes to know how often people were unable to find the correct solution, either by not being able to ask the correct questions or the model providing enough information. Unfortunately, due to an error, we were not able to log one participant.

However, not everyone managed to finish the game in the first round and since we wanted to measure player interaction with the LLM without previous experiences, we decided that we would only consider NPC's results valid, if players encountered them for the first time. This meant that we focused our attention on the first NPC, the noblewoman, which everybody was able to finish in their first playthrough. The second NPC was also a possible candidate, but since she was rather easy to figure out, with only a single person using the command on her, and was also solvable within the first sentence, we thought that the first NPC was a better indicator. In the Table 5.9 it is shown that close to half of all participants had to use the solution command to solve their first interaction with the noblewoman.

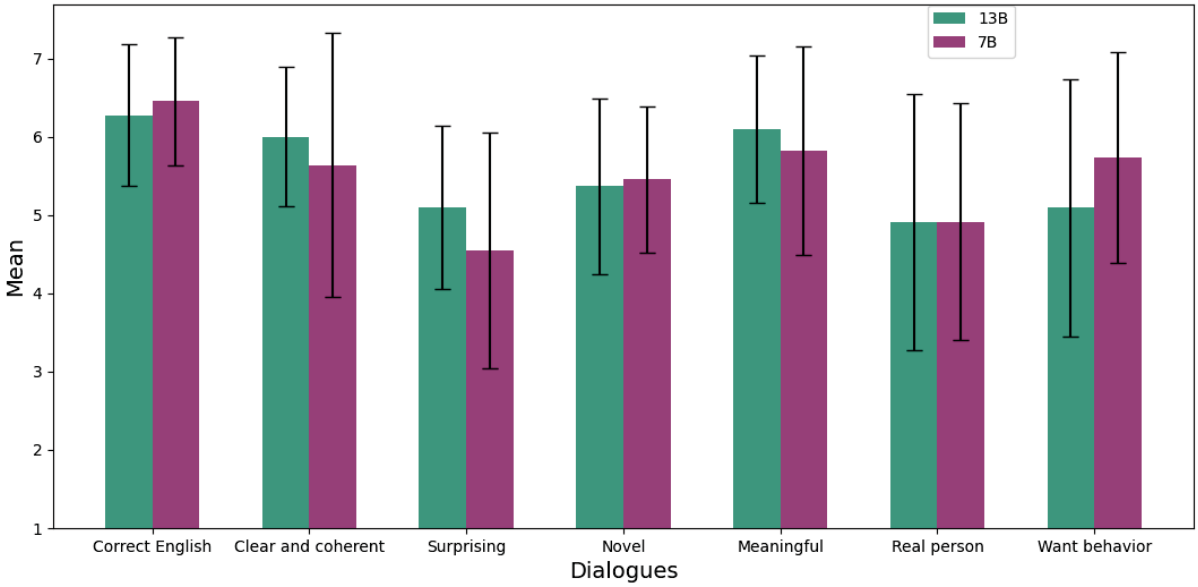


Figure 5.3.: Mean and standard deviation when 7B was shown first.

| Category | Number of people |
|---|---|
| Used the solution command | 11/26 |
| Did not use the solution command | 13/26 |
| Unknown | 1/26 |

Table 5.9.: Number of people using the solution command at least once in their first run to find the correct answer in the first NPC. The "Unknown" is due to a missing log file.

Additionally, to understand the reason why players used the command, we decided to take a look at the interactions with that first NPC and deduce a reason on what the leading reason to utilize it was. As a result we formulated categories and gathered them in Table 5.10. As we can see five of the interactions were mainly because the model invented information which was not part of the game and mislead the players. Therefore, these participants were lead into an unrelated storyline or idle chit chat which did not help progress the story. Sometimes the reason behind this action was that players were formulating the wrong questions, either by being unspecific or inventing new information to convince the NPC , which lead the model to invent information to satisfy the players inquiry.

Furthermore, we found that some people should have successfully progressed the story, but the similarity of the trigger sentence and input was not close enough to be detected. A third category that we identified is, that the NPC seemed to be very uncooperative and tried to change topic or talk about something else. While this behavior is a part of the

| Category | Number of people |
|---|---|
| Invented information | 5/11 |
| Did not trigger action | 2/11 |
| Denies cooperation | 2/11 |
| Unknown reason | 2/11 |

Table 5.10.: Categories on why players used the solution command and their occurrence.

story and intentional, some players might have thought that the conversation would not lead to anything useful. At the end, we were not able to assign any category to the rest and there was no apparent reason for us why these players decided to use the command.

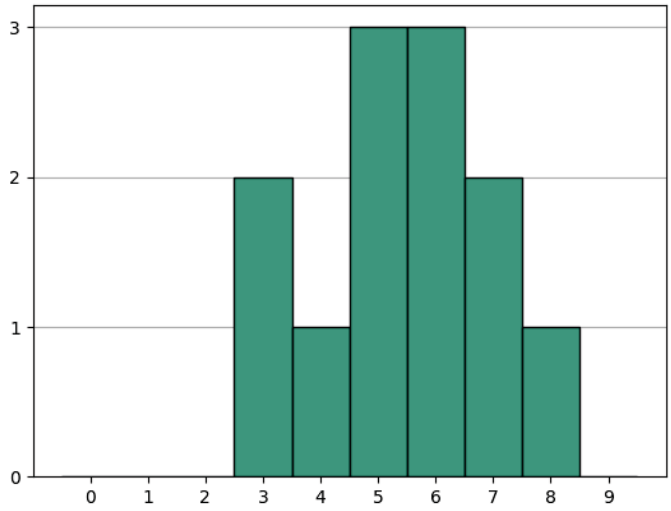


Figure 5.4.: Distribution of how many questions were asked until players used the solution command in the first NPC.

In Figure 5.4 we display how many times participants asked the first NPC a question before they requested the solution using a histogram. As a result, we can see that the players take an average around five to six inputs before they decide to utilize the command, if at all. Although, this measurement might be inaccurate since players are reminded in-game after the sixth user input that they should consider using the solution command soon. Furthermore, since there was a time restriction, players might have felt that they lacked time to interact with the NPC more.

## 5.6. Questionnaire Results

In total, ten women, 15 men and two non-binary people participated in this questionnaire, for a total of 27 people. The ages ranged from 18 to 41, however, the vast majority of

| Category | Handwritten | Llama 2-chat 13B | p-value |
|---|---|---|---|
| Correct English | mean: 5.1111, std: 1.1209 | mean: 4.6296, std: 1.4182 | 0.0470 |
| Clear and coherent | mean: 5.1852, std: 0.6815 | mean: 5.0, std: 1.1094 | 0.4030 |
| Surprising | mean: 4.1111, std: 1.3681 | mean: 3.7407, std: 1.3754 | 0.1011 |
| Novel | mean: 4.3704, std: 1.0432 | mean: 4.3333, std: 1.271 | 0.9566 |
| Meaningful | mean: 4.2963, std: 1.1373 | mean: 4.2963, std: 1.3248 | 0.9343 |

Table 5.11.: Dialogue 1 - Exact values for total mean, standard deviation and p-value between handwritten and Llama 2-chat 13B in the questionnaire in each category.

participants were between the age of 20 to 35. Most participants rated their English proficiency skill five out of five or four out of 5, while only one person rated their skill three out of 5.

Furthermore we collected more data by doing a questionnaire where we showed two different dialogues to the participants, one generated by Llama 13B and one that is human-written but follows a similar dialogue structure to make it comparable to the generated one. Here, we utilized a six point Likert scale ranging from “Strongly disagree”, “Disagree”, “Slightly disagree”, “Slightly agree”, “Agree” and “Strongly agree” without a neutral option. This was done so participants who rushed the questionnaire had to pick an option were they either agreed or disagreed to the category even if it was only slightly. Additionally, we measured if the ratings between the two types of conversations have a significant difference. For this we used a Wilcoxon-signed rank test [Wil45] and formulated the following hypothesis:

$\boldsymbol{H_0}$**:** *There is no difference in ratings between human-written and generated dialogue in this category.* $\rightarrow \mu_{Human} = \mu_{Generated}$

$\boldsymbol{H_1}$**:** *There is a difference in ratings between human-written and generated dialogue in this category.* $\rightarrow \mu_{Human} \neq \mu_{Generated}$

$\alpha = 0.05$

Here, we used the same categories as in the experiments, except without the categories “Want behavior” and “Real person”. The questionnaire consisted of three different pairs of dialogues each had one generated and one human-written text. Firstly, in Figure 5.5 and Table 5.11 which depict results from the first dialogue about a conversation between a noble woman and the main character. We can see that the categories “Correct English” and “Clear and coherent” are the categories with the highest mean score. While the handwritten dialogues all perform slightly better than the LLM, we determined that the “Correct English” is the only significant category in this dialogue where the handwritten conversation outperforms the model.

Secondly, in the second dialogue depicted in Figure 5.6 and Table 5.12 participants dealt

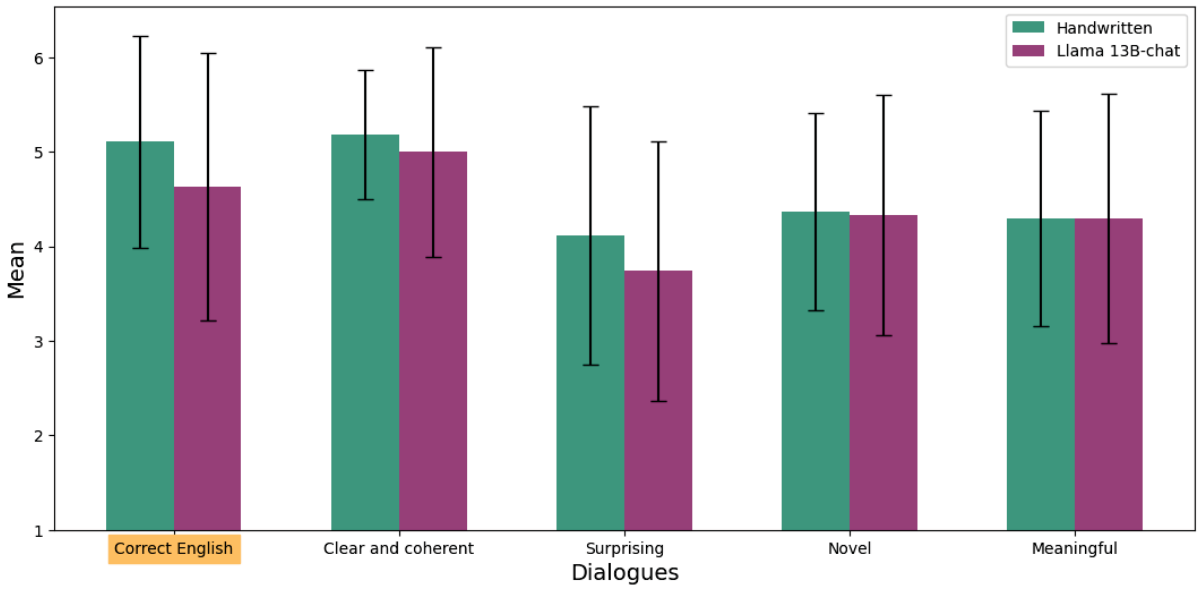


Figure 5.5.: Dialogue 1 - Mean rating between handwritten and generated dialogue in different categories. Marked labels shows significant difference.

with a guard that expressed a stutter. People rated this dialogue, similarly to the nobility conversation but with a much more similar "Correct English" value between the two types. This time, the most striking feature is that the "Meaningful" category achieved a higher rating than the handwritten dialogue, while the "Clear and coherent" category lost ratings in the model conversation. These developments lead to these two categories having a significant difference in humanwritten and model rating, while "Correct English" lost their significance. Analyzing the notes, which each participant gave optionally, we could presume that the stutter in the handwritten dialogue was more convincing than the generated one resulting in improving "Meaningful" and decreasing "Clear and coherent" for the model. Later, we discuss these notes further in a more in-depth qualitative analysis.

| Category | Handwritten | Llama 2-chat 13B | p-value |
|---|---|---|---|
| Correct English | mean: 4.7778, std: 1.1547 | mean: 4.8889, std: 1.086 | 0.5272 |
| Clear and coherent | mean: 4.8148, std: 0.8787 | mean: 4.0741, std: 1.328 | 0.0097 |
| Surprising | mean: 4.1111, std: 1.1209 | mean: 3.6296, std: 1.3909 | 0.08100 |
| Novel | mean: 4.3333, std: 1.2403 | mean: 4.0, std: 1.3009 | 0.0634 |
| Meaningful | mean: 4.8519, std: 0.7698 | mean: 4.2963, std: 1.103 | 0.0413 |

Table 5.12.: Dialogue 2 - Exact values for total mean, standard deviation and p-value between handwritten and Llama 2-chat 13B in the questionnaire in each category.

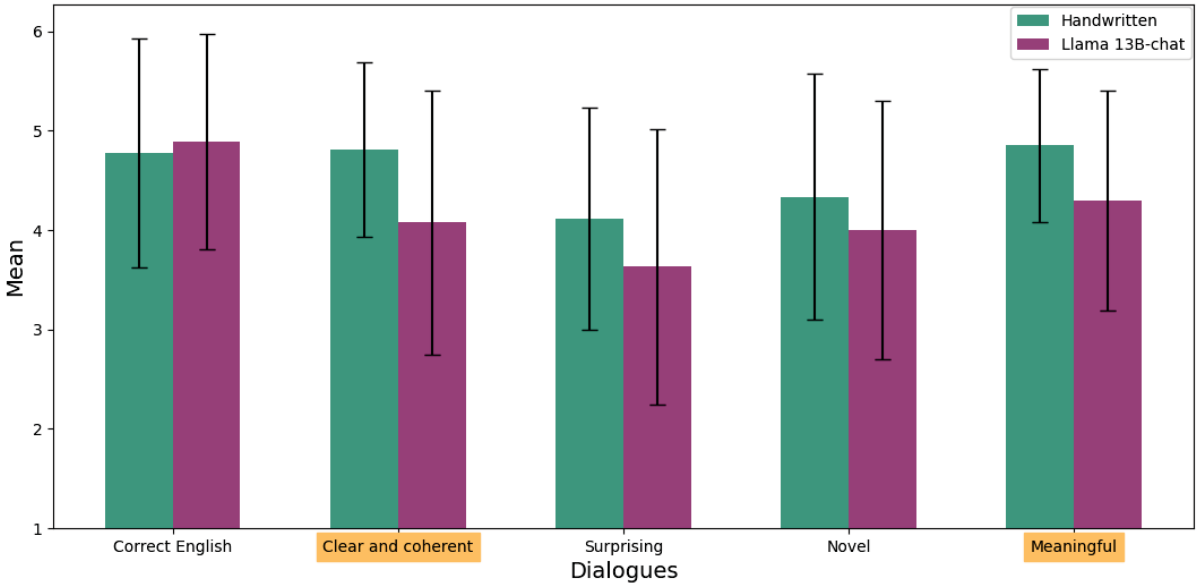


Figure 5.6.: Dialogue 2 - Mean rating of a conversation between handwritten and generated dialogue in different categories. Marked labels shows significant difference.

Lastly, the third dialogue in Figure 5.7 and Table 5.13 follows the conversation between a beggar and their conversation partner. The interesting data in this dialogue is that the "Meaningful" category for the model dialogue was slightly higher than the human-written one. This generated dialogue had a somewhat better performance than the previous ones in "Novelty", "Surprising" and "Meaningfulness". Despite this change, this time there was no significance in difference between any of the categories.

Summarizing all the dialogue ratings in Figure 5.8 and Table 5.14, we can see that the categories "Correct English" and "Clear and coherent" performed the best, while "Surprising" scored the least but was not far behind in comparison to the other bars. Overall, the rating between handwritten and generated conversations performed very

| Category | Handwritten | Llama 2-chat 13B | p-value |
|---|---|---|---|
| Correct English | mean: 4.4815, std: 1.2821 | mean: 4.2222, std: 1.4763 | 0.1354 |
| Clear and coherent | mean: 4.9259, std: 1.0715 | mean: 4.5926, std: 1.2172 | 0.1080 |
| Surprising | mean: 3.7778, std: 1.0127 | mean: 3.963, std: 1.2552 | 0.3750 |
| Novel | mean: 4.2593, std: 1.3183 | mean: 4.5185, std: 1.1222 | 0.3381 |
| Meaningful | mean: 4.6667, std: 1.3587 | mean: 4.8519, std: 1.0991 | 0.5932 |

Table 5.13.: Dialogue 3 - Exact values for total mean, standard deviation, p-value between handwritten and Llama 2-chat 13B in the questionnaire in each category.

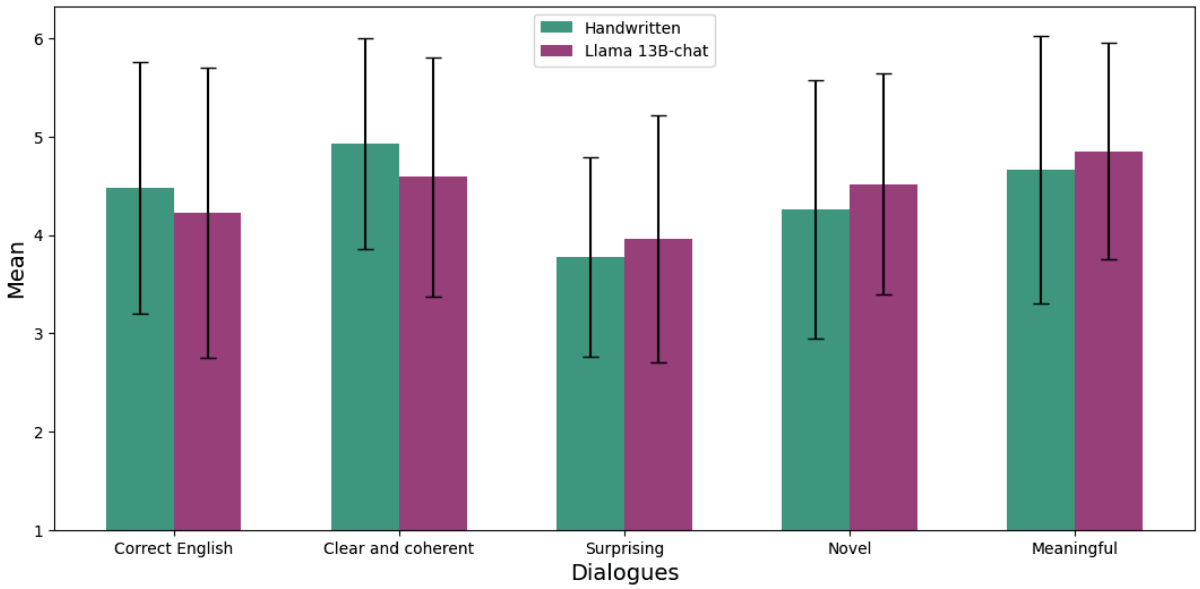


Figure 5.7.: Dialogue 3 - Mean rating of a conversation between handwritten and generated dialogue in different categories.

| Category | Handwritten | Llama 2-chat 13B | p-value |
|---|---|---|---|
| Correct English | mean: 4.7901, std: 1.2012 | mean: 4.5802, std: 1.3497 | 0.0441 |
| Clear and coherent | mean: 4.9753, std: 0.8941 | mean: 4.5556, std: 1.2649 | 0.0021 |
| Surprising | mean: mean: 4.0, std: 1.1726 | mean: 3.7778, std: 1.3323 | 0.1373 |
| Novel | mean: 4.321, std: 1.1919 | mean: 4.284, std: 1.2373 | 0.8511 |
| Meaningful | mean: 4.6049, std: 1.1256 | mean: 4.4815, std: 1.1949 | 0.4257 |

Table 5.14.: Exact values for total mean, standard deviation and p-value between handwritten and Llama 2-chat 13B in the questionnaire in each category.

similarly. However, the categories “Correct English” and “Clear and coherent” were significant enough to reject the null hypothesis.

Following the rating, we also asked participants to pick either the human-written or the model generated dialogue as their preferred conversation without revealing which is which. Additionally, we also formulated a hypothesis to determine if the difference in preference between the handwritten and generated is significant.

$\boldsymbol{H_0}$: *Participants prefer the handwritten and generated dialogue equally.*

$\boldsymbol{H_1}$: *Participants prefer one dialogue over the other.*

$\alpha = 0.05$

When a participant preferred one conversation over another, we gave that conversation a

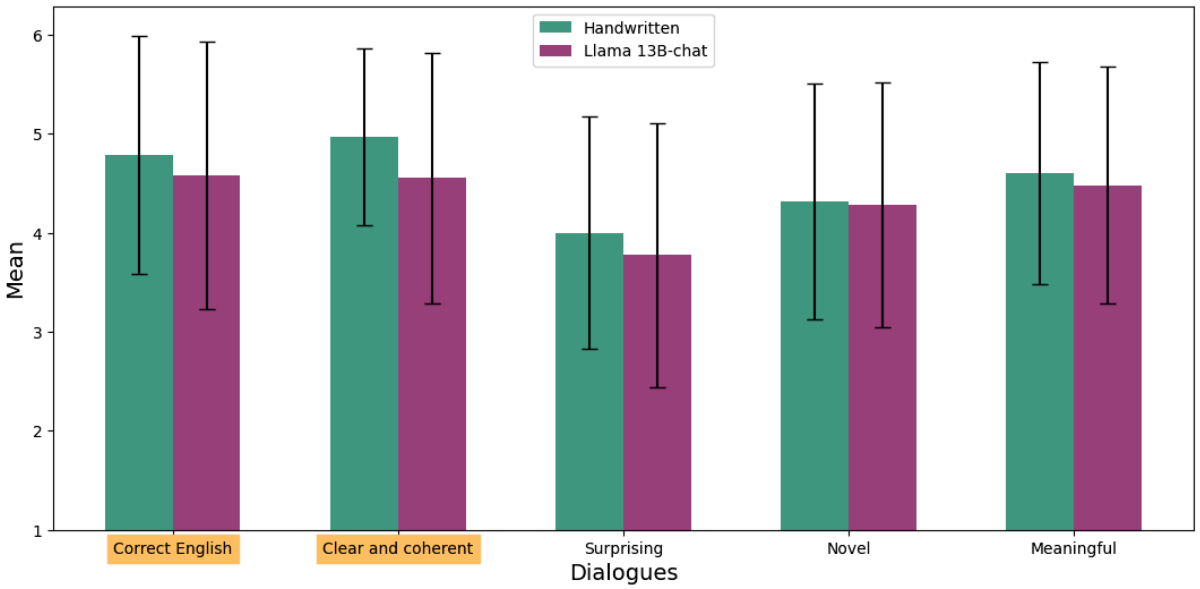


Figure 5.8.: Total mean rating over all conversations between handwritten and generated dialogue in different categories.

score of one and the other a score of zero. As a result, we can see in Figure 5.9, both approval bars seem to be of similar height in the first dialogue, with the handwritten bar being slightly higher, but not enough to reject the null hypothesis. In contrast, we have the second dialogue which shows a great difference between the handwritten approval and model approval, where the handwritten one clearly outperforms the other bar chart. Subsequently, we were able to reject the null hypothesis in this dialogue. Lastly, the third conversation showed that participants preferred the generated dialogue much more than the handwritten one. However, the result was not significant enough to reject the null hypothesis.

Besides evaluating the dialogues on scores we gave the participants an opportunity to provide more insight into why they preferred a text over another, by writing notes on each dialogue. This feedback was optional and gave us valuable information on their process. From the 27 participants approximately 2/3 gave an opinion on each dialogue. Furthermore, to find a better trend, we were able to sort certain words into categories that had the same theme which can be seen in Table 5.15. Note that a single person's feedback could contain multiple words that contributed to different categories, but will not count if it has already been marked in that category.

To start with, the LLM in the first dialogue as shown in Table 5.16 has high ratings in the "Human like behavior" category and "Efficient language". On the other hand, the handwritten conversation has a higher rating in "Appealing language" and "Human like behavior". Additionally, the handwritten dialogue received slightly more negative

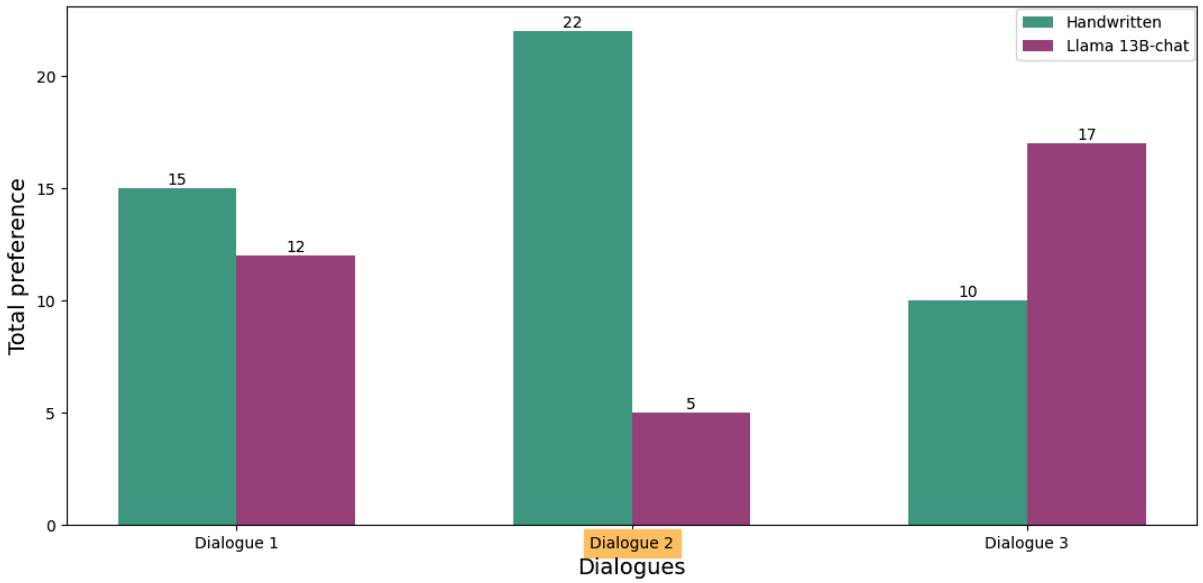


Figure 5.9.: Preference of handwritten and generated text in different dialogues. Marked labels shows that their rating significantly differ from each other.

| Category | Keywords |
|---|---|
| Human like behavior | humanlike, natural, more lively, relatable mood, more personal, real person |
| Less human like behavior | unnatural, less human, unnatural interjection |
| Efficient language | Concise, less exaggerated, less word repetition, clearer, easier to read |
| Inefficient language | too many unnecessary adjectives, too much info, difficult to read |
| Appealing language | engaging, artistic, more endearing, more interesting, fluent, funny, expressive punctuation, less weird, more casual, better stuttering, warmer, more detailed |
| Unappealing language | annoying stutter |
| In-character behavior | more character, better stutter, more timid, unique personality, more in-character, speaking consistently, unique personality, convincing character |
| Out of character behavior | unconvincing speaking style, inauthentic |

Table 5.15.: Feedback from participant notes sorted into categories. The keywords represent which words were sorted into which categories.

feedback in different categories like “Inefficient language” and “Less human like behavior”.

| category | Handwritten | LLM |
|---|---|---|
| Human like behavior | 3 | 6 |
| Less human like behavior | 1 | 1 |
| Efficient language | 1 | 5 |
| Inefficient language | 3 | 0 |
| Appealing language | 6 | 4 |
| Unappealing language | 0 | 1 |
| In-character behavior | 1 | 0 |
| Out of character behavior | 1 | 0 |

Table 5.16.: Dialogue 1 - Occurrence of categories in

| category | Handwritten | LLM |
|---|---|---|
| Human like behavior | 4 | 0 |
| Less human like behavior | 0 | 1 |
| Efficient language | 2 | 0 |
| Inefficient language | 0 | 0 |
| Appealing language | 4 | 0 |
| Unappealing language | 0 | 1 |
| In-character behavior | 3 | 2 |
| Out of character behavior | 0 | 0 |

Table 5.17.: Dialogue 2

As we could see in Figure 5.9 before, the second hand-written dialogue described in the Table 5.17 performed much better than the model and strictly received good feedback in all categories, while the model only had positive responses in the “In-character behavior” category and a few negative feedbacks.

| Category | Handwritten | LLM |
|---|---|---|
| Human like behavior | 2 | 2 |
| Less human like behavior | 0 | 0 |
| Efficient language | 2 | 2 |
| Inefficient language | 0 | 1 |
| Appealing language | 1 | 3 |
| Unappealing language | 0 | 0 |
| In-character behavior | 1 | 4 |
| Out of character behavior | 1 | 0 |

Table 5.18.: Dialogue 3

Lastly, the third dialogue in Table 5.18 stands out in the sense that the two versions have similar category ratings except in the “Appealing language” and “In-character behavior”

categories. They both also received negative feedback, the handwritten one had one in "Out of character behavior", while the LLM received one in "Less human like behavior".

| Category | Handwritten | LLM |
|---|---|---|
| Human like behavior | 9 | 8 |
| Less human like behavior | 1 | 2 |
| Efficient language | 5 | 7 |
| Inefficient language | 3 | 1 |
| Appealing language | 11 | 7 |
| Unappealing language | 0 | 2 |
| In-character behavior | 5 | 6 |
| Out of character behavior | 2 | 0 |

Table 5.19.: Total - Occurrence of categories in

When we sum up the performance of all three dialogues in the Table 5.19, we can see that people most often give feedback on the categories "Human like behavior" and "Appealing language". And while the other categories differ only by one or two more mentions, "Appealing language" sticks out with almost having more mentions in the handwritten conversation which also had no mention that it was "Unappealing language".

In addition to the preference, we also asked the questionnaire participants to identify which text they think is written by a human. In this task, they were also asked to justify their choice, which was not optional. Furthermore, we analyzed the relationship between the preference of the conversation and the identification of the human texts. Moreover, we grouped the data into wether the participants preference and human text identification matched each other or if they differed. These findings were collected in the Table 5.20 for people who chose the same dialogue for preference and identification, and in Table 5.21 where the choice of preference and identification differed from each other.

| Preference | Identified human text correctly | Dialog 1 | Dialog 2 | Dialog 3 | Sum |
|---|---|---|---|---|---|
| Handwritten | Yes | 12 | 19 | 8 | 39 |
| LLM | No | 8 | 4 | 13 | 25 |
| Sum of matching preference & identification | | 20 (74,07%) | 23 (85,19%) | 21 (77,78%) | 64 (79,01%) |

Table 5.20.: Relationship of dialogue where preference and identification of the human text are the same

In the Table 5.20, we can see that in total 79% of all participants liked the text they thought was written by a human. On the other hand, only 21% of all participants preferred the text that they thought was not written by a human. As a result, it is possible that people who are of the opinion that one is the human text, wether or not

they correctly identified it, will often also prefer it.

| Preference | Identified human text correctly | Dialog 1 | Dialog 2 | Dialog 3 | Sum |
|---|---|---|---|---|---|
| Handwritten | No | 3 | 3 | 2 | 8 |
| LLM | Yes | 4 | 1 | 4 | 9 |
| Sum of matching preference & identification | | 7 (25,93%) | 4 (14,81%) | 6 (22,22%) | 17 (20,99%) |

Table 5.21.: Relationship of dialogue where preference and identification of the human text are different

# 6. Discussion

In this section we talk about the data that we have found in our evaluation and how we can interpret the result. Furthermore, we will also discuss the limitations of our experiment and questionnaire.

We have seen in the experiment that the overall performance of different sized models are very similar in their ratings in different categories. Interestingly, when looking at the total mean rating, no category has a rating below 4.5 out of 7, most of the categories even range from five to 6. It suggests that the models are performing well when it comes to generating dialogue texts. However, one has to keep in mind that the rating is not solely due to the model generation, since fetching context and information is also a very important part of generating appropriate response. Thus, the ratings here is the combination of an LLM with a RAG system that is able to detect and fetch relevant context and not the performance of a pure LLM alone.

Especially interesting are the categories "Real person" and "Want behavior", which asks about if the text behaves human like and if participants would like that behavior in other games respectively. Here, we can see that "Real person" performs worse than all other categories, however, they are still rated fairly good. Another observation is that this category has a lower performance than "Want behavior", which indicates that just because the model is not considered to be a real human, people still appreciate the dialogue style in NPC dialogue.

Since Csepregi [Cse23] executed similar experiments with an LLM in NPCs, we compared our results to their data. There they used the same metric as van Stegren et al. [vSM21] suggested with a baseline and a context-aware measurement. Since the baseline is still working with context and their additional context-aware version contains interactions that we do not have in our experiments, we decided to compare our data with their baseline data. Additionally, the only category in which our metric slightly differs is the "Creativity" category, which we named "Meaningful". Since we are unsure if these two categories would be interpreted the same, we want to point out that these two might not be comparable to each other. In total, the mean seems to be almost identical in the category "Correct English", while "Surprising" and "Meaningful" only differs by 0.2 to 0.3 points. However, the standard deviation appears to be consistently lower by at least 0.3 points in all of the five categories, even ranging up to 1.1 points in "Surprising". This indicates that our ratings were more often around the mean than in the experiments of Csepregi, suggesting a more consistent rating in all categories. Furthermore, the category "Clear and coherent" also had similar results depending on the model that was used. The

13B model achieved approximately 0.5 points more in mean and 0.7 points less in their standard deviation than their baseline mean, while the 7B model achieved approximately 0.3 points more in mean and 0.2 less in their standard deviation. This was the only time model size mattered in a category when comparing to the baseline of Csepregi. The true outlier was "Novel", where our mean is approximately 1.1 points higher than their found value and 0.6 lower in standard deviation. The reason why the performance is better in this category, while being similar in all other categories is unclear and would need further investigation.

However, we could also gather from the responses that the 7B model behaved more lively and more unpredictable, often adding unexpected symbols and hashtags. Additionally, based on the feedback, the model seems to mainly differ in shorter responses and more frequent generation errors, like not providing necessary information, hallucinating or being unable to play an NPC anymore. We conclude, based on this information, that this behavior shows that the smaller model may act more erratically in comparison, which helps it play characters with more expressive language. On the other hand, we have the 13B model which is described as having longer dialogue texts and having less random symbols. Furthermore, there were no mentions that the bigger model withheld important information or not behaving like an NPC, in contrast to the 7B model. Moreover, many players expressed that the accents stood out and either increased the playing experience or were being too exaggerated. As a result, we think that the 13B model is more detailed and adheres more to instructions, at the cost of being slightly less convincing in impersonating characters.

In addition to the ratings and the qualitative analysis of the texts, we also looked at how many players were able to play the game without using the solution command. We found that about half of the players had to use it in their first NPC interaction. Since we did not inquire for what motivated them to use the command, we can only assume the reasons behind it. However, from what we could interpret, we identified that often players were unable to continue due to being frustrated by not being able to find the solution phrase. Often the model generated unnecessary or misleading information which can and will lead players into a different direction than intended. The reason for this might be that players ask about information which we did not create and as a result has to be invented by the model. Since one of the intentions of incorporating LLMs in NPC dialogue generation is to make dialogue dynamic and make interactions in the world more lively by filling in information gaps, inventing information that is not inhibiting the progress or contradicting the story is intended. However, we could gather that the model was inventing information that was misleading and was often the reason some player barely made any progress and used the solution command instead.

To minimize this behavior, we suggest that characters are equipped with more context and knowledge of their story, without providing secret information. Since our game is based on unlocking information, we decided to guarantee withholding certain pieces of information. Because sometimes the models did not receive sufficient texts as context to answer the user question, the model had to cope with only having access to basic

information in their character prompts, which might have lead to hallucinations about new quests and people. However, we observed that not all information was provided completely correctly. Sometimes, the LLM omitted or changed crucial descriptions of an object or a situation that was part of the intended story, which could have impaired the gameplay. In our experiment, most of the times this did not lead to anything game breaking, since the essence of the information was still conveyed sufficiently to progress the story.

What did work well was the use of RAG where we could add information quickly without fine-tuning the model and risking time and resources. The RAG text embedding model often accurately detected trigger phrases correctly and allowed our game when players figured out the solution. Unfortunately, in a few cases inputs were correct but were just under the threshold of the solution phrases, inhibiting the progress of some players. This was due to players writing more than just the trigger phrase or it written too differently than what was expected, which resulted that the similarity score was below the necessary activation score. For example, a player answers with "I will ensure that your secret is safe with me and no harm will come to you." when the trigger phrase was "I can keep your secret." it might not be similar enough to trigger the solution. However, it might have been good enough if the player omitted the second part after the "and". An adjustment to the activation score might increase false positive inputs or decrease false negative inputs in other phrases. The solution for this problem is to have a better algorithm where correct answers can be detected more accurately. Nonetheless, in our opinion, the use of RAG is a very integral part in our game and should be included in future game development where NPCs with in-game world knowledge are involved

In the questionnaire results, the presented Llama 13B model performed very similarly when compared to the handwritten dialogues. So much so, that from the overall scores none of the categories were statistically significant. Since we presented three dialogues of different characters overall, we also measured their individual rating and found that the second dialogue has performed significantly better than the other two. Because we let participants voluntarily comment on their decision on how they perceived the conversations, we could determine that many of the comments were related to it being more natural and clear than the generated one. Some people criticized that the stuttering of the LLM was annoying or unnatural, while the human-written text exhibited more human stuttering and was clearer. On the other hand, we can also see that the preference in dialogue three is bit more on the LLM side than on the human text, but they do not differ significantly enough like in dialogue 2.

Analyzing the notes of the participants, we can see that in total, handwritten and LLM dialogues have very similar feedback. Even the category "Human like behavior" was very evenly distributed between the LLM and handwritten category. Some noticeable difference is that the handwritten text feedback included more criticism in different negative categories, but also had a slightly better rating in appealing language.

Overall, we can see that the performance of NPC text can vary depending on how the

character speaks and behaves. This leads us to believe that when creating a new NPC character that they should be evaluated on a case to case basis since small changes in character e.g. a stuttering in the speaking patterns, can skew the preference of players. As we have seen in the experiment, some generations might even be too exaggerated to be real and would break the conversation flow. This problem might be improved by identifying the shortcomings in speaking patterns, finding good training data and fine-tune specific areas to improve e.g. stuttering.

To put these result in context, we compared them to the results from Stegren et al. [vSM21], after converting our six point Likert scale to a seven point Likert scale using the formula $((\text{rating} - 1) \cdot \frac{6}{5}) + 1$ on each rating [JL20].

Stegren et al. have done similar experiments concerning quest descriptions, which were character statements about a quest they had to offer. Even though this was not a multi turn conversation like we had, it should be similar enough to be comparable to our data, since they use the same metric. As before, the only category that differed from theirs was the category "Creative" which was "Meaningful" in our questionnaire.

As a result, we have found that our handwritten result were consistently lower than their handwritten rating in all categories. "Correct English" and "Surprising" had the lowest values with around 1.1 points lower than theirs. The most similar rating was the "Novel" category with only a difference of 0.13. However, our generated values performed better in comparison in most categories, except "Correct English" by 0.24 and "Surprising" by 0.72. Otherwise, our ratings were slightly better by approximately 0.3 to 0.6 in all other LLM groups. Furthermore, by comparing which categories were significant, we found that "Correct English" and "Clear and coherent" were also the significant categories in their experiments. However, we could not confirm that their third significant category "Novelty" was also significant for our analysis.

The reason why our handwritten data were rated worse, might be that Stegren et al. used actual game quest description with professional writers rather than writing them ourselves. However, our model generated slightly better text than their GPT-2 fine-tuned model, which might indicate that our generalized non fine-tuned Llama 2 can at least keep up with older specialized fine-tuned ones. This could mean that replacing fine-tuned model with newer non-fine tuned models might be possible for game development when updating a game, which could save further cost and time in maintenance.

Furthermore, we have found an interesting relation between identification of human texts and the preference for one of the conversations. In the second analysis of the questionnaire data, we noticed that many people have picked a preference between the dialogues and when asked to identify the human text, subsequently also often picked the preferred dialogue.

This leads us to believe, that participants might favor the conversations that they believe are human written and not necessarily the better quality of the texts. This can potentially mean that the model do not need to generate perfect responses like written

fictional characters, but it could mean that including small human mistakes and other human quirks may improve the perception of the characters. Although this insight is not conclusive by solely analyzing this data, it might help to improve the quality of models in games for the future.

Additionally, even though the metric of Stegren et al. [vSM21] worked well, it did mostly focus on analyzing text's quality in the assumption that it would automatically be the better received texts. However, based on the data in the experiments and questionnaire, participants seem to also enjoy not entirely correct responses, which might have been perceived as more human instead of written by a static machine. Additionally, the metrics used, did not show a much different rating between the first dialogue and the second dialogue, even though the human text was heavily preferred by participants. It suggests that the metric is not capturing some hidden factor in the rating and might need improvement. Thus, measuring the human likeness of the model could be a good additional category to evaluate the performance in role-playing NPCs in future work.

In conclusion, to answer the second research question concerning "Do users prefer LLM or human written dialogue and for which reasons.", we can say that models have improved so much that in many cases they are not significantly better or worse than human written conversations. At most the handwritten texts might have a slight advantage in the use of correct English and coherency than in the generated conversations.

Additionally, the data indicates that participants who think a text is human, will most likely also pick it as their preferred text. From the questionnaire we can gather, that to evaluate some NPC conversations and personalities it will require an individual evaluation, since small quirks in personality might not be emulated well enough and may lead to a disturbance in the game experience. Furthermore, we found that the measurement of Stegren et al. [vSM21] does not sufficiently detect the preference of players and would require further improvement. This finding may be related to Touvron et al. [TMS+23] where they noted that human evaluation is noisy and subjective when it comes to generated text, since changes in the prompts can lead to different results and evaluations.

A big limitation in both our questionnaire and our experiments was, that we have mainly asked people who where in the Faculty of Computer Science of the University of Vienna. This sampling naturally selects younger, German speaking people who have a higher education, which could skew the ratings due to not having diverse enough participants. Future studies should consider a larger and more diverse sample through different age groups, gender, education and language background. However, we are confident that trends identified in this research provide a valuable starting point for future projects.

Additionally, the metric we used to measure the performance might not be sufficient to accurately quantify their performance. We have found that many of the results were not very different between humand-written and generated conversations, however, even though one of the dialogues had a clear preference by participants, the metric that we used was not sufficient to reflect their preferences.

Another issue that we encountered is the experiment design. Even though we tried to explain the game mechanics verbally, from the feedback we received, many players struggled with them. As a result, some participants were not prepared to play the game and in turn rated the second playthrough better than the first one, which we did account in the findings. However, verbal instructions did not seem to help everybody understand the mechanics, thus a solution could be to first let them play a short tutorial or visually show them examples on how to interact with the system.

On the other hand, researcher sometimes had to help players recall information that NPCs mentioned which were integral to the continuation of the story. In one or two sessions a trigger was not activated correctly even though the system should have accepted it. In these cases we encouraged the players to type a similar phrase again. This might have influenced a few experiments, since otherwise they would have used the solution command or have given a lower rating.

Additionally, time pressure and the presence of a researcher may have influenced the behavior of the participants. Having only 15 minutes to play each playthrough while a person is sitting besides them reminding them of their time, does not reflect real life gaming, where players do not have to rush through the game. Some players even expressed that the game was a little short, indicating that a more accurate rating could be achieved when given more time.

To summarize, future experiments should improve the game design and a more clear experiment protocol to ensure more consistent and accurate results.

Fine-tuning was not used due to the bad performance of the trained model, however, a more effective fine-tuned NPC model could increase the ratings of the game. The reason why the trained model was performing so badly might be because of the LIGHT training data [UFK+19] that was directly crowd sourced from role-playing players. We noticed that the model had a worse spelling than in the untrained model, which leads us to believe that the training data contained spelling mistakes that our model learned. To mitigate this issue, a more careful selection of the training data with a better RLHF approach would be needed.

Another limitation can be found in our questionnaire. In order to make human-written text comparable to the generated one, we had to abstract the LLM text and reformulated it based on keywords. However, a human may otherwise have not formulated the conversation in this way and would have answered the questions differently. This method could possibly affect the handwritten score and make it more similar to the generated one.

# 7. Conclusion

In the first part of our research, we investigated to what extend Large Language Models (LLMs), combined with Retrieval-Augmented Generation (RAG), generate dialogue that is suitable for non-player character interactions. Therefore, we implemented a textadventure game in which players could interact with characters in the story through typing. In order to generate the dialogues, we utilized Llama 2-chat [TMS+23], a fine-tuned version of Llama 2 which possesses chatbot skills. Additionally, we implemented a Retrieval-Augmented Generation system that provided context to the generator, by identifying which character knowledge was relevant to the input and adding it to the prompt.

In an experiment, we asked participants to play our game two times, once with Llama 2-chat 7B and once with Llama 2-chat 13B. We found, that both models were very similar in their ratings and were both suited for use in NPC dialogues, however, analyzing the feedback notes of the participants we could determine that while the 13B model was longer and more precise in its generation, the 7B model generated shorter and more chaotic output.

Overall, players rated the performance of both models very well, with a mean score of around 5.9 out of seven points when asked if they would like this NPC behavior in other games and around five out of seven when asked if they think the NPCs talked like a real person. This leads us to believe that the use of models combined with a RAG system has great potential, although issues, like hallucinating information and sometimes not recognizing the correct answer, need to be addressed before a full fledged game can be developed.

In the second part of our research, we set out to answer the question if users prefer text generated by LLMs or human-written text and for which reasons. Here, we designed a questionnaire that compared Llama 2-chat 13B to handwritten dialogue without them knowing which text is generated. Overall, human-written text performed better in the use of correct English and in clearness and coherency, but otherwise participants did not rate one text significantly better than the other. The only exception was the second dialogue where human-written text was significantly preferred, due to the character having a more natural stutter than in the generated text. This leads us to be optimistic that character behavior has to be evaluated on a case-by case basis, since small insufficiently trained speaking patterns, could lead to less preferred generations.

Furthermore, we analyzed the preference and found that the majority of participants preferred the dialogue that they thought was written by a human, regardless if that was

the case. Using this information, we have found that a better metric has to be researched, in which a rating on how much players perceive the text as human-written could be a good indicator of the player preference.

There are various ways to improve the project for future studies. For example, we have only used Llama 2-chat which was fine-tuned to be a chatbot. However, newer models with chatbot abilities might yield a better result than the one we used. For future work, we suggest to use different newer models and compare their results.

Additionally, a more efficient RAG system could be implemented, where the model has a better ability to find all relevant documents. This can be done by using a newer text or document embedding model than mixedbread-ai [LSKL24] that we used. Another suggestion to improve the efficiency of the RAG system is to save all textual data already tokenized in the database. Even better would be to embed these texts beforehand and save them by using a vector database so the retriever can efficiently query document data.

Furthermore, future work on this subject could also be to fine-tune the model using better NPC dialogue. Since there are many games with NPC conversation, handwritten by professionals, some of them could be used to improve the text generation. However, some of these data could be licensed and can not be used for training.

Another idea is to create a custom dataset using other LLMs which are curated and picked using human feedback. Subsequently, fine-tuning could also be combined with RLHF and instruction fine-tuning techniques in order to have better instruction following capabilities.

An additional way to further improve generation quality is to use hyperparameter optimization before fine-tuning. The parameter in our configuration like temperature, repetition quality and others might be too high or too low to have an optimal quality.

Furthermore, there is need to look into research and how a better metric for generated NPC conversations can be constructed. Since prompts and model configuration can influence the quality of the output, a better benchmark is required that does not rely on how well a prompt is constructed.

# Bibliography


[Aga19] Abien Fred Agarap. Deep learning using rectified linear units (relu), 2019.

[AWK+23] Trevor Ashby, Braden K Webb, Gregory Knapp, Jackson Searle, and Nancy Fulda. Personalized quest and dialogue generation in role-playing games: A knowledge graph- and language model-based approach. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, New York, NY, USA, 2023. Association for Computing Machinery.

[BMR+20] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[Bod09] Margaret Boden. Creativity in a nutshell. *Think*, 5:83–96, 09 2009.

[Cse23] Lajos Matyas Csepregi. The effect of context-aware llm-based npc conversations on player engagement in role-playing video games, 2023.

[DCLT19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[DPHZ23] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023.

[dvg25] Wikimedia Commons contributors dvgodoy. File:transformer, one encoder-decoder block.png, 2025. Last access: 29.09.2025.

[ECK+25] Kenneth Enevoldsen, Isaac Chung, Imene Kerboua, Márton Kardos, Ashwin Mathur, David Stap, Jay Gala, Wissam Siblini, Dominik Krzemiński, Genta Indra Winata, Saba Sturua, Saiteja Utpala, Mathieu Ciancone, Marion Schaeffer, Gabriel Sequeira, Diganta Misra, Shreeya Dhakal, Jonathan Rystrøm, Roman Solomatin, Ömer Çağatan, Akash Kundu, Martin Bernstorff, Shitao Xiao, Akshita Sukhlecha, Bhavish Pahwa, Rafał Poświata, Kranthi Kiran GV, Shawon Ashraf, Daniel Auras, Björn Plüster, Jan Philipp Harries, Loïc Magne, Isabelle Mohr, Mariya Hendriksen, Dawei Zhu, Hippolyte

Gisserot-Boukhlef, Tom Aarsen, Jan Kostkan, Konrad Wojtasik, Taemin Lee, Marek Šuppa, Crystina Zhang, Roberta Rocca, Mohammed Hamdy, Andrianos Michail, John Yang, Manuel Faysse, Aleksei Vatolin, Nandan Thakur, Manan Dey, Dipam Vasani, Pranjal Chitale, Simone Tedeschi, Nguyen Tai, Artem Snegirev, Michael Günther, Mengzhou Xia, Weijia Shi, Xing Han Lù, Jordan Clive, Gayatri Krishnakumar, Anna Maksimova, Silvan Wehrli, Maria Tikhonova, Henil Panchal, Aleksandr Abramov, Malte Ostendorff, Zheng Liu, Simon Clematide, Lester James Miranda, Alena Fenogenova, Guangyu Song, Ruqiya Bin Safi, Wen-Ding Li, Alessia Borghini, Federico Cassano, Hongjin Su, Jimmy Lin, Howard Yen, Lasse Hansen, Sara Hooker, Chenghao Xiao, Vaibhav Adlakha, Orion Weller, Siva Reddy, and Niklas Muennighoff. Mmteb: Massive multilingual text embedding benchmark. *arXiv preprint arXiv:2502.13595*, 2025.

[Fre99] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[Hip20] Richard D Hipp. SQLite, 2020.

[HSW+21] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

[JL20] Jeff Sauro Jim Lewis. How to convert between five- and seven-point scales, 2020.

[JM25] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, with Language Models*. 3rd edition, 2025. Online manuscript released August 24, 2025.

[KMH+20] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020.

[LHI+23] Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for "mind" exploration of large language model society, 2023.

[LLY+23] Cheng Li, Ziang Leng, Chenxi Yan, Junyi Shen, Hao Wang, Weishi MI, Yaying Fei, Xiaoyang Feng, Song Yan, HaoSheng Wang, Linkang Zhan, Yaokai Jia, Pingyu Wu, and Haozhen Sun. Chatharuhi: Reviving anime character in reality via large language model, 2023.

[LPP+21] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.

[LR05] Erich Leo Lehmann and Joseph P Romano. *Testing statistical hypotheses*. Springer, 2005.

[LSKL24] Sean Lee, Aamir Shakir, Darius Koenig, and Julius Lipp. Open source strikes bread - new fluffy embeddings model, 2024.

[MLWF21] Andrea Madotto, Zhaojiang Lin, Genta Indra Winata, and Pascale Fung. Few-shot bot: Prompt-based learning for dialogue systems, 2021.

[Sha51] C. E. Shannon. Prediction and entropy of printed english. *The Bell System Technical Journal*, 30(1):50–64, 1951.

[Sha20] Noam Shazeer. Glu variants improve transformer, 2020.

[SLDQ23] Yunfan Shao, Linyang Li, Junqi Dai, and Xipeng Qiu. Character-llm: A trainable agent for role-playing, 2023.

[SLP+23] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023.

[SMR23] Murray Shanahan, Kyle McDonell, and Laria Reynolds. Role-play with large language models, 2023.

[SW65] Samuel Sanford Shapiro and Martin B Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3-4):591–611, 1965.

[TLI+23] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[TMS+23] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[TZL+23] Chen Feng Tsai, Xiaochen Zhou, Sierra S. Liu, Jing Li, Mo Yu, and Hongyuan Mei. Can large language models play text games well? current state-of-the-art and open questions, 2023.

[UFK+19] Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. Learning to speak and act in a fantasy text adventure game, 2019.

[VSD+23] Sebastian Vincent, Rowanne Sumner, Alice Dowek, Charlotte Blundell, Emily Preston, Chris Bayliss, Chris Oakley, and Carolina Scarton. Personalised language modelling of screen characters using rich metadata annotations, 2023.

[vSM21] Judith van Stegeren and Jakub Myśliwiec. Fine-tuning gpt-2 on annotated rpg quests for npc dialogue generation. In *Proceedings of the 16th International Conference on the Foundations of Digital Games*, FDG '21, New York, NY, USA, 2021. Association for Computing Machinery.

[VSP+23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[WBZ+22] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners, 2022.

[Wil45] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics bulletin*, 1(6):80–83, 1945.

[XGDM23] Canwen Xu, Daya Guo, Nan Duan, and Julian McAuley. Baize: An open-source chat model with parameter-efficient tuning on self-chat data, 2023.

[ZSW+20] Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences, 2020.

# A. Appendix

| Theme | Associated Notes |
|---|---|
| Characters behavior fit their role | Notes 1: 4, 8, 18, 22, 25, 26, Notes 2: 8, 13, 22 |
| Characters behavior did not fit their role | Notes 1: 2, 8, 15, 18, 22, Notes 2: 18 |
| Characters were enjoyable | Notes 1: 15, 24, Notes 2: 8, 12, 18, 24 |
| Characters were less enjoyable | Notes 1: 8, 10, 15, Notes 2: 16, 23 |
| Characters were unique | Notes 1: 11, 12, Notes 2: 11, 13, 25, 26 |
| Dialogue contained errors | Notes 1: 17, Notes 2: 2, 8, 11, 17, 18 |
| Miscellaneous | Notes 2: 3, 10, 15 |

Table A.1.: All collected notes from the experiment grouped into categories. Number correspond to row in raw experiment data. Notes 1 and Notes 2 describe in which run they wrote their opinions.

| Theme | First | Second |
|---|---|---|
| Characters behavior fit their role | 4, 18 | 2, 8, 22 |
| Characters behavior did not fit their role | 18 | - |
| Characters were enjoyable | - | 8, 12, 24 |
| Characters were less enjoyable | - | - |
| Characters were unique | 11 | 25, 26 |
| Dialogue contained errors | - | 2, 8, 17 |
| Miscellaneous | - | 15 |

Table A.2.: All collected notes from 7B grouped into categories. Number correspond to row in raw experiment data.

| Theme | First | Second |
|---|---|---|
| Characters behavior fit their role | 8, 22, 25, 26 | 13 |
| Characters behavior did not fit their role | 2, 8, 15, 22 | 18 |
| Characters were enjoyable | 24 | 18 |
| Characters were less enjoyable | 8, 10, 15 | 16, 23 |
| Characters were unique | 12 | 11, 13 |
| Dialogue contained errors | 17 | 11, 18 |
| Miscellaneous | - | 3 |

Table A.3.: All collected notes from 13B grouped into categories. Number correspond to row in raw experiment data.

| Model | First | Second |
|---|---|---|
| Llama 7B-chat | 3, 11, 16, 21, 23 | 7, 8, 12 |
| Llama 13B-chat | 2, 10, 17, 20, 22, 24, 26 | 4, 9, 14 |
| Preference due to order 7B | 13, 18 | 5, 6, 15, 25, 27 |
| Preference due to order 13B | | 19 |

Table A.4.: Preferences of 7B and 13B excluding the preferences due to order. Number correspond to row in raw experiment data.