Procedural Quest Generation Using LLM

Caden White
Oregon State University
Corvallis, Oregon, USA
Whitecad@oregonstate.edu

Abstract

Quest are a staple to video games and video game design, seeing use in a variety of video game from Role playing Games to First Person Shooters. Quest have become very stagnant over the years leading players to become unengaged with the game and feel as if they are completing a to-do list rather than playing a video game. Due to advancements in machine learning, but more specifically Large Language Models, procedural quest generation is becoming a new topic of interest. In this paper I will use a pretrained GPT model, train it on a RPG quest dataset. Using this new model I aim to evaluate it using quantitative metrics. Through this I aim to explore this new and emerging field and determine the strengths and weakness of both the model and the field.

1. Introduction

Throughout the recent stretch of gaming and video game development quests have become a go-to mechanic for game developers. Quests can be seen in a range of games from your favorite Role Playing Games(RPGs) like Skyrim and Destiny to First person shooters(FPS) like Call of Duty and Fortnite. Quests are used as a game mechanic for a variety of reasons, such as in RPG's they are used to give the player direction and encourage them to explore, whereas in games like First Person Shooters they retain engagement by encouraging players to play in new and exciting ways. At the end of these quests the player is then usually rewarded with some in-game currency, or experience points. A major issue among these quests though is eventual through the repetition of similar quests, most notable in the FPS section of games, the quests become repetitive and boring, eliminating the original purpose and making the game overall feel sluggish and more like a to-do list.

With the advancement of natural language processing(NLP) and Large Language Models(LLMs), many areas of research have become possible, including automatic quest generation. In this paper I aim to explore the possibilities of automated quest generation using a transformer based architecture and fine tuning it on a RPG quest dataset. Analyzing the models performance through quantitative measures as well as outright observations we can gain insight into the model's strengths and weaknesses, determining its fit and future in the role of video game development. The paper is organized as follows. Section 2 describes related work and the history of procedural quest generation. Section 3 introduces the proposed methods and describes the implementation details. Section 4 presents an evaluation of the model. Section 5 discusses the potential future of procedurally generated quests and offers concluding remarks.

2. Related Works

While the field of procedural quest generation using LLMs has only found recent success, quest in video games have had a notable history alongside other solutions outside LLMs. Here is a brief overview of related work starting with the origin of quests and leading up to recent techniques. While quests have had a long history in gaming it is very unclear as to who first implemented it. Some credit this to familiar titles like, "the legend of zelda"[4] or classic titles in, "Ultima IV: Quest of the Avatar"[4]. While they are credited as some of the first to incorporate quests they appear very different than they do today. These appear in the form of optional content with no clear description or quest management system seen commonly today. Quest has since developed since then due to games like Skyrim, giving the player a dedicated menu option to quest, with descriptions and objectives[5]. These quests were pre-programed, usually using a branching system where each player's choice and how the game would respond were baked into the game allowing the player some agency in their choices but only within the limited responses. Up until recently quests remained a tool of RPGs, as they are a great tool to develop the story and guide the player. Fortnite, a widely popular game, back in 2018 released the battle pass, a place where by completing quests the player could get experience for the battle pass to unlock rewards[6], opening up quests for FPS.

These quests utilize a template based system having some basic template and rotating in certain parts i.e. "Destroy [X] ..." or "discover ..."[7].

Around the late 2010's procedural quest generation research started with machine learning and planning based approaches. In the paper presented by Lima and Furtado[8], they take a genetic algorithm approach with automated planning. In the paper they describe a model that is composed of two subsystems, an offline quest generator and a game manager. The offline quest generator is composed of the genetic algorithm and the quest planners. The genetic algorithm which is used to efficiently search the game space full of items, NPC's enemies and places and determine a good sequence of tasks. The quest planner takes in the quest generated by the genetic algorithm and determines if it is a good enough quest. Then once the offline generator has created the quest, the game manager takes the quest and the current world state and presents the quests to the player throughout the game.Other similar approaches also were developed at the time like the paper presented by Breault et al.[9] which describes Creation Of Novel Adventure Narrative (CONAN) another procedural quest generator that uses a planning approach. Lastly Lima et al.[10] take another similar approach using genetic algorithms and Automated planning, yet they extend the model to work on branching quests, a staple RPG quest design.

Finally the most recent developments in the field of procedural generated quests comes from the rise in LLMs and NLP.One unique approach comes from Ashby et al.[1] who utilizes both a knowledge graph and a language model to tackle the issue. They take a very technical approach, taking input from the player about a task or goal, utilizing a knowledge base of the world state, generating a quest, and then utilizing a LLM to generate the response as well as key features of the quest to the player.Another take Värtinen et al.[3] took a very similar approach, utilizing openAI's GPT-2 and GPT-3 models and fine tuning them on a publicly available dataset of quest descriptions. These projects display the current work towards procedurally generated quests, upheld by the recent success of LLMs. This project also aims to capitalize on the rise in success of these LLMs and provide a simple and realistic evaluation of these models and the future.

3. Methodology

3.1. Dataset

The dataset used for both training and evaluating the model is the NPC Dialogue RPG Quests dataset from Hugging Face. The dataset was structured into three sections one for the title of the quest, the objective of the quest, and the NPC responding dialogue. This dataset contains approximately 24,000 samples making it much larger than other examples. The dataset was separated on a 99.8% training and a 0.2% testing split giving us a total split of 24,790 training samples and 50 testing samples.This was done as any larger was too much strain on the training system. Before training the dataset was preprocessed using the hugging face datasets library. First the dataset was tokenized using the GPT-2 Tokenizer, this took each sample, changing each sample to a string of the format, "Title: [title] Objective: [objective] Text: [text]". After this, the tokenizer, tokenized the string, or took words or subwords and converted them into integers representing that token in the tokenizer. On top of this it applied padding and truncation to maintain a maximum token length of 256. Once this was done it was then converted into a pytorch tensor making it usable by hugging face's trainer package.

3.2. Model architecture

The model architecture that is used is Open AI's GPT-2 transformer-based architecture[11,12]. The architecture used in this specific model utilizes 117 million parameters, 12 attention heads,12 hidden layers and 768 dimensional states. This specific model is classified as a causal language model(CLM) where it uses transformers[13], taking in input tokens and output tokens and then predicting what the next token is most likely to be. To use this model I import the hugging face transformers library giving me access to a wide range of transformer models but also the tool to train and evaluate the model.

3.3. Training Hyperparameters

Training was done using the trainer class, a tool from the hugging faces transformer library. The trainer class allowed me to easily set training parameters for the model. The parameters used to train the model, were a batch size of 4, 5 epochs, a learning rate of $5 * 10^{-5}$, and adamW as the optimizer. AdamW[14] is an optimization algorithm used in Deep learning models to help the model "learn". AdamW is used because it is one of the most commonly used optimization algorithms for LLMs but on top of that provides better generalization and less overfitting. This is due to the addition of not only momentum and RMSprop but AdamW specifically separates the weight decay from the gradient update leading to better generalization and less overfitting.

4. Quantitative Evaluation

After Training evaluation was done many metrics were used to determine the performance of the model. The following metrics were used in this evaluation.

4.1. BLEU

Bilingual Evaluation Understudy(BLEU) is a widely used metric for LLMs as it is useful in determining similarity between the reference quest and the model generated

quest. the BLEU formula[15] is described as $BLEU = BP \times exp(\sum_{n=1}^{N} w_n log(p_n)$ where N is the maximum n-gram size, w_n is the weight of each n-gram precision score typically uniform, p_n is the precession score for n-grams, and BP is the Brevity penalty which prevents short generation from being unfairly rewarded. BP is defined as $BP = \begin{cases} 1, c \geq r \\ e^{1-r/c}, c \leq r \end{cases}$ where c is the length of the candidate translation and r is the effective reference corpus length. Another evaluation metric that is possible using BLEU is self-BLUE. self-BLEU when calculating the BLEU score uses the other generated text compared to the generated texts reference. This is our most beneficial evaluation metric as it allows us to see the uniqueness and diversity in the generated quests.

4.2. ROUGE

Recall-Oriented Understudy for Gisting Evaluation(ROUGE) is also another commonly used evaluator for LLMs. ROUGE generates a score representing the similarity between generated and referenced text or quest. In our evaluation we use three different ROUGE scores, ROUGE-1, ROUGE-2, and ROUGE-L. ROUGE-1 calculates a score based on the overlap of unigrams or single words/tokens. This is done via a simple formula of ROUGE-1 $= \frac{\text{\# of matching unigrams}}{\text{Total unigrams}}$. ROUGE-2 is very similar were insteaed of calculating matching unigrams, we go based off of match bigrams, or sequences of tokens, which is calculate like ROUGE-2 $= \frac{\text{\# of matching bigrams}}{\text{Total bigrams}}$. Lastly there is ROUGE-L which is a little more different where it calculates its score based on the longest common subsequence(LCS), which is represented like, ROUGE-L $= \frac{\text{LCS}}{\text{Reference Text Length}}$.

4.3. Distinct-n

Distinct-n is the last evaluation metric used on the dataset. Distinct-n measures the unique number of n-grams in the generated text.In the evaluation Distinct-1 and distinct-2 were used. Distinct-1 measures the unique unigrams compared to the total unigrams in the generated text, deriving the equation, Distinct-1 $= \frac{\text{unique unigrams}}{\text{Total unigrams}}$. Distinct-2 follows the pattern messuring the unique bigrams among the total, giving us Distinct-2 $= \frac{\text{unique bigrams}}{\text{Total bigrams}}$

4.4. results

Using the 50 samples we retained from the training set, at the end of training we ran all of the evaluation metrics ending with the scores showing in table 1.

Evaluation Method	Score
BLEU	0.184
Self-BLEU	0.131
ROGUE-1	0.53
ROUGE-2	0.21
ROUGE-L	0.44
Distinct-1	0.396
Distinct-2	0.85

Table 1: Evaluation Scores from testing.

Starting off with the BLEU score, it is clear that the model had trouble with similarity between the model's prediction and the reference text and reviewing Appendix A there are major issues with the model. We can clearly see issues especially towards the beginning which is most likely because the model has no input to guide it. Looking at self-BLEU though we can see some positive results as he self-BLEU score indicates that among the predicted quest they were fairly diverse which was a main goal of this experiment. Moving to the ROUGE scores, there are also some positive take always here in that the ROUGE-1 and ROUGE-L produced decent results displaying that there was good vocab similarity and good semantic similarity, but it did have some issues producing connected vocab. Lastly there is the Distinct section. The Distinct-1 section produced okay results displaying that three was some variety among unigrams but it could do better. On the other hand Distinct-2 displayed that there was quite a diversity among bigrams. These results display that the model performed at an okay standard, and after reviewing appendix A,a example prediction and reference, its clear that the model left a little more to be desired.

5. Conclusion

The recent developments in LLMs and NLPs has created more possibilities in the realm of not only procedural generated quest, but procedural generated content as a whole, and this project encapsulates that. This project in comparison to the projects listed in the related works section are not as technical nor as big, but i think through this it has displayed that this field isn't a short sighted endeavor. Through the use of a weaker GPT model and a small dataset, this paper was able to produce decent results displaying decent cohesion between predictions and references, displayed by appendix A but also can generate great output with some help from the user displayed by appendix B. Overall, this project validates the feasibility of using smaller LLMs and modest datasets for quest generation, while also highlighting the potential for improvement through larger models, more diverse training data, and refined evaluation techniques. It

serves as a stepping stone for future work in automated content creation for games, where combining procedural generation with human creativity could lead to more immersive and dynamic experiences.

6. references

[1]Ashby, Trevor & Webb, Braden & Knapp, Gregory & Searle, Jackson & Fulda, Nancy. (2023). Personalized Quest and Dialogue Generation in Role-Playing Games: A Knowledge Graph- and Language Model-based Approach. 1-20. 10.1145/3544548.3581441.

[2]E. S. Lima, B. Feijó, A. L. Furtado, Hierarchical Generation of Dynamic and Nondeterministic Quests in Games, in: Proceedings of the 11th International Conference on Advances in Computer Entertainment Technology, 2014, Article N. 24. https://doi.org/10.1145/2663806.2663833.

[3]S. Värtinen, P. Hämäläinen and C. Guckelsberger, "Generating Role-Playing Game Quests With GPT Language Models," in IEEE Transactions on Games, vol. 16, no. 1, pp. 127-139, March 2024, doi: 10.1109/TG.2022.3228480.

[4]"What Was the First Video Game to Have Side Quests?" Quora, www.quora.com/What-was-the-first-video-game-to-have-side-quests. Accessed 22 Mar. 2025.

[5]Scrolls, Contributors to Elder. "Quests (Skyrim)." Elder Scrolls, Fandom, Inc., elderscrolls.fandom.com/wiki/Quests_(Skyrim). Accessed 22 Mar. 2025.

[6]Wiki, Contributors to Fortnite. "Season 2." Fortnite Wiki, Fandom, Inc., fortnite.fandom.com/wiki/Season_2#Battle_Pass. Accessed 22 Mar. 2025.

[7]"Daily Quests" Fandom, fortnite.fandom.com/wiki/Daily_Quests. Accessed 22 Mar. 2025.

[8]Lima, Edirlei Soares de. "Procedural Generation of Quests for Games Using Genetic Algorithms and Automated Planning" 7 Nov. 2019, www.inf.puc-rio.br/wordpress/wp-content/uploads/2019/11/www.inf.puc-rio.br/wordpress/wp-content/uploads/2019/11/196293.pdf. Accessed 22 Mar. 2025.

[9]Vincent Breault, Sébastien Ouellet, Jim Davies,"Let CONAN tell you a story: Procedural quest generation," Entertainment Computing, Volume 38, 2021,100422, ISSN 1875-9521, https://doi.org/10.1016/j.entcom.2021.100422.

[10]Edirlei Soares de Lima, Bruno Feijó, Antonio L. Furtado, "Procedural generation of branching quests for games," Entertainment Computing, Volume 43, 2022, 100491,ISSN 1875-9521, https://doi.org/10.1016/j.entcom.2022.100491.

[11]Radford*, Alec. "Language Models are Unsupervised Multitask Learners" 14 Feb. 2019, cdn.openai.com/better-language-models/cdn.openai.com/better-language models/language_models_are_unsupervised_multitask_learners.pdf. Accessed 22 Mar. 2025.

[12]8 June 2018, cdn.openai.com/research-covers/language-unsupervised/cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf. Accessed 22 Mar. 2025.

[13]"Attention Is All You Need" 6 Dec. 2017, arxiv.org/abs/1706.03762. Accessed 22 Mar. 2025.

[14]"Decoupled Weight Decay Regularization" 4 Jan. 2019, arxiv.org/abs/1711.05101. Accessed 22 Mar. 2025.

[15]Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL '02). Association for Computational Linguistics, USA, 311–318. https://doi.org/10.3115/1073083.1073135

Appendix A. Generated vs reference example

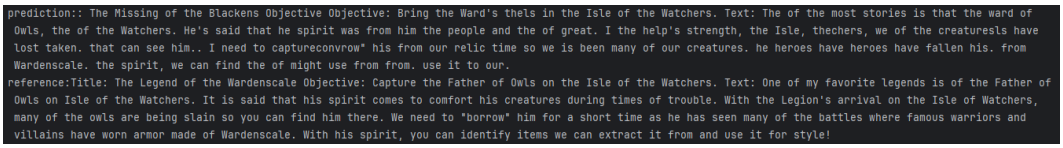

Figure 1: A sample generated quest output and its corresponding reference.

Appendix B. Supported Model examples

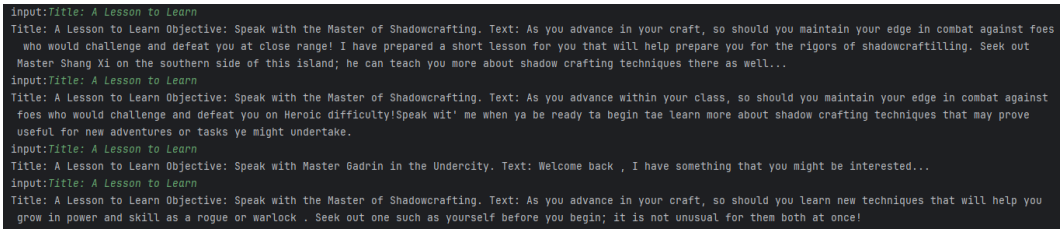

Figure 2: Examples generate from the model that had prior input, ”Title: A Lesson to Learn”

Appendix C. Source Code

White, Caden (2025) ”Procedural Quest Generation Using LLM” source code (Version 1.0) https://github.com/whitecad1228/Quest-RPG-LLM.