AI Dungeon Master for D&D 5e: Report

Overview

Building upon the basic prototype made previously the added components include generating a character based on player input, extracting and enhancing details from a character's backstory, and generating a vivid setting description based on the input.

Character Stats Generation

This section generates the player’s character stats using a dice roll system commonly found in D&D.

- **Roll Stats**: roll_stat(), simulates a 4d6 roll, discarding the lowest value, and summing up the remaining three dice to generate a stat value.
- **Generate Stats**: The function generate_stats(race, char_class) calculates the stats of the character (Strength, Dexterity, Constitution, Intelligence, Wisdom, and Charisma) using race-specific and class-specific modifiers. The modifiers for various races and classes are predefined.

Character Creation

The create_character() function facilitates the creation of a character by prompting the user for several pieces of information:

- **Race**: The available races are Elf, Dwarf, Human, Halfling, Dragonborn, Tiefling, Gnome, Half-Elf, Half-Orc, Tabaxi, Aasimar, and Goliath.
- **Class**: The available classes are Wizard, Fighter, Rogue, Cleric, Paladin, Ranger, Bard, Warlock, Monk, Barbarian, and Druid.
- **Alignment**: The alignment options are Lawful Good, Neutral Good, Chaotic Good, Lawful Neutral, True Neutral, Chaotic Neutral, Lawful Evil, Neutral Evil, and Chaotic Evil.

Additionally, the player is also asked to provide a character name, backstory, and appearance. The character's stats are then generated and returned as a dictionary containing all of the details.

Extracting Details From Backstory

The extract_details_from_backstory() function utilizes the SpaCy NLP library to extract important details from the character's backstory. These details include:

- **Locations**: Geographic locations or specific settings.
- **Objects**: Objects, events, or significant groups that might appear in the setting.

The SpaCy model (en_core_web_sm) processes the backstory to identify named entities and noun tokens that fit the categories of locations and objects.

Reconstructing Prompt From Extracted Details

The construct_prompt_from_backstory() function enhances the character's backstory by appending the extracted locations and objects to provide a richer context. The final backstory contains the details of the locations and key elements in the setting.

Setting Generation

The generate_dynamic_setting() function constructs a prompt based on the character's name, race, class, stats, and backstory. This prompt is fed into the model to generate a setting description. The function performs the following:

- It uses the character's attributes (strength, dexterity, etc.) and backstory as part of the prompt to ensure the generated setting aligns with the character's background.
- The generated setting is a narrative description, intended to establish the environment in which the character's adventure will take place.

Game Setup

The game setup involves the combination of character creation, setting generation, and the Dungeon Master's narrative introduction. Here, the following steps occur:

- The character is created by calling create_character().
- The character's backstory is processed to extract locations and objects.
- A dynamic setting is generated based on the character's information.
- The game begins with the Dungeon Master introducing the setting and the character's arrival in the unfolding adventure.

Challenges Faced

After integrating user input for character details, we tried to get better setting generation by extracting key details from the input like geographic locations, objects by using SpaCy to identify named entities and noun tokens that fit these categories. Using these details, we redefined the prompt to get better response. But the model still fails sometimes to incorporate the details given by the user and gives irrelevant settings that deviates from the character's backstory.

Further Improvement

For further improving our model in the upcoming days we intend to:

- **Combat Mechanics:** Integrate combat mechanics to simulate battles
- **Finetuning:** Training the model on the datasets we have collected in the beginning

- **Prompts:** The prompts can be made even better to make sure that we extract better responses from the model