Leveraging Large Language Models for Spell-Generation in Dungeons & Dragons

Elio Musacchio, Lucia Siciliani, Pierpaolo Basile, Giovanni Semeraro
Department of Computer Science, University of Bari Aldo Moro (Bari, Italy)
elio.musacchio@phd.unipi.it, {name.surname}@uniba.it

Abstract

Dungeons&Dragons (*D&D*) is a classic tabletop game with a 50-year history. Its intricate and customizable gameplay allows players to create endless worlds and stories. Due to the highly narrative component of this game, *D&D* and many other interactive games represent a challenging setting for the Natural Language Generation (NLG) capabilities of LLMs. This paper explores using LLMs to generate new spells, which are one of the most captivating aspects of *D&D* gameplay. Due to the scarcity of resources available for such a specific task, we build a dataset of 3,259 instances by combining official and fan-made *D&D* spells. We considered several LLMs in generating spells, which underwent a quantitative and qualitative evaluation. Metrics including BLEU and BertScore were computed for quantitative assessments. Subsequently, we also conducted an in-vivo evaluation with a survey involving *D&D* players, which could assess the quality of the generated spells as well as their adherence to the rules. Furthermore, the paper emphasizes the open-sourcing of all models, datasets, and findings, aiming to catalyze further research on this topic.

Keywords: Generative Artificial Intelligence, Large Language Model, Text Generation, Dungeons&Dragons

1. Introduction

In tabletop role-playing games, *Dungeons & Dragons* (*D&D*) is a timeless classic, captivating players with its immersive storytelling, strategic gameplay, and boundless possibilities. Central to the experience of *D&D* is the use of spells, which enable players to wield magical forces, shape reality, and overcome challenges within the game world.

D&D operates within a framework of rules, facilitating structured gameplay while allowing for creativity and improvisation. Players assume the roles of characters with distinct abilities, embarking on adventures guided by a Dungeon Master (DM) who orchestrates the narrative and adjudicates the rules. One of the most integral aspects of character abilities in *D&D* is casting spells, which encompass a vast array of effects ranging from elemental manipulation to healing and illusion.

Traditionally, spell-casting in *D&D* has relied on predefined lists of spells published in rule books, with players selecting spells for their characters based on predefined criteria such as character class, level, and available spell slots. However, creating new spells or expanding the existing repertoire has mainly been relegated to game designers or enthusiasts, often requiring extensive manual effort and expertise.

In recent years, the emergence of Large Language Models (LLMs) powered by artificial intelligence has revolutionized various domains, including content generation and creative writing. These models are trained on vast corpora of text data and demonstrate remarkable capabilities in understanding and generating human-like text.

This paper explores the potential of leveraging open-source LLMs to generate spells in *D&D*, aiming to augment the creative possibilities within the game. By harnessing the generative power of LLMs, players and game designers can unlock a wealth of spell variations, improving the game with greater depth, novelty, and customization options. In particular, our work proposes a methodology for integrating LLMs into the spell generation process. We investigate the feasibility of automating spell creation by fine-tuning recent open-source LLMs on a dataset of spells manually generated by *D&D* enthusiasts. Furthermore, we explore the implications of employing LLMs for spell generation in *D&D*, including the quality of generated spells and the change in performance by varying the number of model parameters. Ultimately, this paper contributes to the intersection of artificial intelligence and tabletop gaming, demonstrating how advanced language models can enrich the creative processes inherent in games like *D&D*, fostering innovation in gaming communities.

The paper is structured as follows: Section 2 provides an overview of related work, while the methodology is deeply described in Section 3. The evaluation and results are discussed in Section 4.

2. Related Work

Large Language Models (LLMs), built upon the Transformer architecture (Vaswani et al., 2017), undergo extensive training processes facilitated by immense datasets. Their sheer magnitude enables them to encapsulate and process many linguistic

Games and NLP @LREC-COLING-2024, pages 61–69

patterns and structures. Notably, LLMs excel not only in mastering downstream tasks but also in their ability to generate text, marking a significant milestone in the field of Natural Language Processing.

Creating narratives for games is of particular interest since it not only requires text coherence but also a more complex structure of the generated story. More specifically, for tabletop games like *D&D*, there are many aspects to take into account, e.g. the setting of the story, the characters, and the specific game state, thus requiring the system to expose Language Generation, Language understanding and planning abilities (Callison-Burch et al., 2022). Moreover, while LLMs have proved to be able to generalize across tasks for which they were not directly trained (Wei et al., 2021; Kojima et al., 2022), they still need huge corpora for the fine-tuning step. This represents an additional challenge within the *D&D* realm: in fact, despite being a game boasting 50 years of history since its first publication and counting an enormous amount of information online, there is a limited number of well-structured datasets related to this topic.

Due to the complexity of *D&D*, many researchers have approached this problem along different lines of research. For example, in (Callison-Burch et al., 2022), the authors focus on the dialogue-based nature of the game given its turn-based system. They fine-tune an LLM, i.e. the 64B LaMDA language model (Thoppilan et al., 2022), on data crawled from *D&D* Beyond Play-by-Post[1]. In (Rameshkumar and Bailey, 2020), the authors focus on text summarization techniques applied to *D&D*. They also built the Critical Role *Dungeons & Dragons* Dataset (CRD3), composed of dialogues extracted from a show called Critical Role, along with abstractive summaries extracted from the Critical Role Fandom wiki. The FIREBALL dataset (Zhu et al., 2023), was instead obtained by collecting 25,000 *D&D* sessions with information about the game state and then used to evaluate the ability of LLMs (i.e. GPT3) in predicting the next game command, given the utterances from the last turns.

Another relevant aspect in any *D&D* campaign is represented by spells, which are magical incantations or formulae that characters can cast to achieve various effects within the game world. Spells are a fundamental aspect of gameplay, allowing players to shape the course of their adventures by adding depth, strategy, and excitement to their story. Regarding works more similar to our contribution, we firstly refer to the work proposed in (Newman and Liu, 2022). Here, the authors focus on spell generation, using three different models: one based on a simple N-Gram model, one based on LSTM (Hochreiter and Schmidhuber, 1997), and the final one based on GPT-2 (Radford et al., 2019). The authors make use of a dataset composed of 3,062 spells, which is obtained by combining the Kaggle dnd-spells dataset[2] and player-made spells from the *D&D* wiki[3]. In detail, the authors use 3,012 randomly chosen spells from this dataset as training data, while the other 50 represent the test data. Results obtained using the BLEU and BERTScore evaluation metrics show that GPT-2 is the best-performing model. Given the high rate with which new LLMs are developed and released, we wanted to explore how newer models can perform the spell generation task.

3. Methodology

As discussed previously, we aim to study the effectiveness of using LLMs for *D&D* spell generation.

More specifically, we are interested in assessing the performance of newer LLMs available at the state-of-the-art and checking how much the number of parameters of the model affects the generation process. This aspect is particularly interesting since the available data for this task is limited.

In fact, even if there are several datasets that have been publicly released about *D&D*, the amount of official *D&D* spells is not even in the thousands. Because of this, we are interested in analyzing how well the same type of model architecture performs with respect to the number of parameters and how well it maintains the generalization capabilities for which LLMs are known.

It is important to note that, throughout this work, we employ a pipeline similar to the one used in (Newman and Liu, 2022), which used GPT-2 for this task. This choice was made to allow for comparison with those already available at the state-of-the-art and to glean insights into the effectiveness and performance of our methodologies. Finally, to the best of our knowledge, the source code and final models adopted in (Newman and Liu, 2022) are not publicly available. In contrast, we have made all our code [4], models [5], and dataset [6] available. We firmly believe in transparency and accessibility in research, especially in complex and challenging domains.

3.1. Dataset

For the dataset, we follow and extend the process described in (Newman and Liu, 2022). First, we

[1] https://www.dndbeyond.com/forums/d-d-beyond-general/play-by-post

[2] https://www.kaggle.com/datasets/mrpantherson/dndspells

[3] https://www.dandwiki.com/wiki/

[4] GitHub Repository

[5] HuggingFace Collection

[6] Dataset

download the dnd-spells dataset (introduced in section 2) from Kaggle containing all *D&D* 5th-edition spells and obtain a total of 554 official spells. Each spell in this dataset is characterized by a total of 12 fields, summarized in Table 1.

Field	Description
name	Name of the spell
classes	Classes that can learn the spell
level	The level of the spell (this is separate from the player's level), 0 is a cantrip
school	What type of magic the spell is
cast_time	How long it takes the character to cast the spell
range	How far the character can be from the target of the spell
duration	How long the spell lasts
verbal	Boolean - spell requires a verbal incantation
somatic	Boolean - spell requires a precise hand motion
material	Boolean - spell requires a physical object which is consumed
material_cost	What type of material is consumed when casting the spell
description	The effect casting spell creates

Table 1: Fields in the *D&D* spell dataset.

To further increase the amount of data, we enhance this first set by scraping the "Spell" section of the *D&D* Wiki site[7], which contains community-made content. We retrieve all data for all spells of the main section, while ignoring spells that fall under the categories *"April Fool's Spells"* and *"Incomplete Spells"*, which are more likely to be not fully described or not following the rules. The complete list of spells also contains official *D&D* spells, but the page associated to such spells contains a copyright disclaimer instead of the data, in such cases we directly skip the page. After scraping, we obtain an additional dataset consisting of $3,287$ spells. However, analysis of the dataset revealed that there were many formatting and data quality issues with these spells. Therefore, we perform both a filtering and a pre-processing step to guarantee that the quality of the spells of this additional set matches the official ones. The filtering steps are the following:

- Removed spells instances not having one of the following attributes: *level*, *school*, *duration*, *casting time*, *range*;
- Removed spells instances which did not properly contain the required *components* to cast the spell. In fact, each instance should explicitly state whether the spell requires *Verbal*, *Somatic* or *Material* (in the case of material, we extract the required components). All of this is done through a regex matching operation;
- Each spell is also associated with a list of character classes that can actually learn the spell. Since the content-made spells may also contain content-made classes, we decided to remove the classes that referred to a fan-made class for each spell. The list of official classes was retrieved from the official *D&D* Beyond site[8]. If a spell did not have any classes left after filtering the list, the spell instance was removed.

As part of the spell processing pipeline, we employ the following steps:

- We identify instances where the scraper output contains the number of votes a spell has received on the wiki. This information is irrelevant to our analysis and can potentially distort our results. Therefore, we use a regex matching operation to remove this information;
- When extracting information about spells from the wiki, the level of the spell and the school of magic it belongs to are often combined into a single string of text (e.g. *"9th-level necromancy"*). In order to make use of this information, we need to separate the level and school into their own distinct values;
- We extract the materials needed by spells with a *Material* ("M") as one of their components. This is done by using a regex matching operation and assuming that such information is between brackets.

After filtering and processing the dataset, we gathered a total of 2,705 instances, which were then merged with the Kaggle dataset to obtain our final dataset of 3,259 instances. Finally, in contrast to the approach presented in (Newman and Liu, 2022), where the authors selectively retain a subset of spell features, focusing primarily on elements deemed crucial for the task, such as *name* and *description*, our methodology encompasses all features extracted through the scraping process. The rationale behind this decision is that the LLM should be able to learn the patterns underlying the values for these features. For instance, it should recognize that the presence of a chant in a spell's description correlates with the "Verbal" component. Thus, we decided to leverage all the features available within the dnd-spells dataset. This approach allows the LLM to capture and learn the relationships that can appear among different spell attributes.

[7] https://www.dandwiki.com/wiki/5e_All_Spells

[8] https://www.dndbeyond.com/classes

3.2. Models

For our analysis, we focused on using decoder-only families of models. In particular, the models taken into account for our evaluation are the following:

- **GPT-2** (Radford et al., 2019) *Generative Pre-Trained Transformer*: released by OpenAI in 2019. We use this family of models as a baseline against which we measure our experimental outcomes. This decision not only allows for a comparison of our results but also facilitates the reproducibility of findings established in prior research, such as those documented in (Newman and Liu, 2022). The models chosen for the evaluation are: `gpt2`, `gpt2-medium`, `gpt-large`, and `gpt-xl`;
- **OPT** (Zhang et al., 2022) *Open Pre-Trained Transformer Language Models*: released by Meta AI in 2022, the main appeal of this family of models is that, to the best of our knowledge, this is the only family of models providing many pre-trained checkpoints having different number of parameters (e.g. 125m, 350m, 2.7b, 6.7b, ...). This is very useful for our purposes, as many of the modern models are released with only a limited selection of parameter counts. The models chosen for the evaluation are: `opt-125m`, `opt-350m`, `opt-1.3b`, `opt-2.7b`, `opt-6.7b`, and `opt-13b`;
- **LLaMA 2** (Touvron et al., 2023) *Large Language Model Meta AI*: released by Meta AI in 2023, the available checkpoints have 7b, 13b and 70b parameters. We also consider these models to compare the performance of OPT and GPT-2 (a relatively older family of models) to the performance obtained by these more powerful and recent models. The models chosen for the evaluation are: `llama2-7b` and `llama2-13b`.

3.3. Training details

For the train-test split, we adhere to the methodology outlined in (Newman and Liu, 2022). Specifically, we employ random sampling to extract 50 instances from the complete dataset, which are then designated as the test set, using the rest as the train set.

We perform full-parameter tuning using DeepSpeed ZeRO 3 [9] for parallelism. We train the models using from 1 to 2 nodes equipped with 4 64GB NVIDIA A100 GPUs. The training procedure was performed with a fixed number of 3 epochs over the entire train set as proposed in (Newman and Liu, 2022). We used the following hyperparameters when they were not specified in the original work:

- **maximum sequence length**: 512; each model has a different maximum length. This value was chosen after tokenizing the dataset with each tokenizer and studying the number of tokens required for each spell to avoid truncation
- **learning rate**: 0.00005
- **effective batch size**: 128
- **optimizer**: adamw (torch)
- **packing**: true
- **weight decay**: 0.0001
- **warmup ration**: 0.0
- **seed**: 42

Furthermore, we format the input data following the Alpaca instruction format (Taori et al., 2023) and perform instruction-tuning with a pre-defined instruction: *"Write a spell for the 5th edition of the Dungeons & Dragons game."*. The idea is that the models could be then further fine-tuned for *D&D* related tasks, such as character or scenario creation. An example of a formatted train instance is shown in Table 2.

Below is an instruction that describes a task. Write a response that appropriately completes the request. ### Instruction: Write a spell for the 5th edition of the Dungeons & Dragons game. ### Response: Name: Light Blood Transfusion Level: 1 School: Necromancy Classes: Sorcerer, Warlock, Wizard Casting time: 1 Action Range: Touch Duration: Instantaneous Components: V, S Description: You lay your hand upon a creature, transfering your blood to the target. You spend one or more hit dice, and the touched creature regain hit points equal to number rolled in the hit dice + your spellcasting ability. If the creature is poisoned, you can use a hit die to end the condition on the target, instead of restoring its hit points.

Table 2: Example of prompt.

4. Evaluation

As presented in Section 3.3, the test set consists of 50 instances. Therefore, we perform a quantitative evaluation procedure and also a qualitative one with actual *D&D* players.

[9] https://www.deepspeed.ai/

4.1. Quantitative Evaluation

For quantitative evaluation, we refer to (Newman and Liu, 2022) and use BLEU (Papineni et al., 2002) and BertScore (Zhang* et al., 2020) as evaluation metrics.

BLEU is a widely used metric to evaluate the quality of text generated by LLMs, although it was originally conceived to measure the quality of Machine Translation models. The BLEU score measures the alignment of n-grams between the candidate translation and the reference translation, counting the number of matched n-grams to ascertain translation quality. A higher number of matches typically signifies a superior candidate translation.

BertScore is a powerful metric that is used as an evaluation tool for natural language generation. It is based on the pre-trained BERT contextual embeddings, which are highly effective in capturing complex language structures. Unlike traditional metrics based on n-grams (e.g. the aforementioned BLEU) that can be limited in their ability to capture long-range dependencies, BertScore computes similarity based on contextualized token embeddings. This allows it to effectively capture distant dependencies and similarities, thus producing a more accurate evaluation.

For both metrics, we use the implementation provided in the "evaluate" library by the HuggingFace team [10]. We adopt these two metrics since they are the same ones the authors of (Newman and Liu, 2022) used in their experimental setting. To compute both metrics, a reference sentence must be provided. While (Newman and Liu, 2022) considers the first 40 tokens of the entire spell (therefore starting from the "Name" attribute), we decided to keep 20 tokens after the "Description" attribute, which is the last tag of the entire spell. Our choice is motivated by the assumption that the description is the central aspect to evaluate since it is the part of the spell that contains more text.

Finally, we do not use tokens for the splitting since we are comparing different models and tokenizers. Therefore, we split based on words, keeping the first 20 words as the reference sentence. Words are identified by splitting based on multiple white spaces and using some heuristics to account for punctuation. Results of the quantitative evaluation in terms of BLEU and BertScore are summarized in Table 3.

Analyzing the results, we observe that the best performance are obtained by models with the largest number of parameters (i.e. `llama2-13b` and `opt-13b`). An exception is represented by `llama2-7b` which achieves the best BertScore (recall), even though the differences are quite narrow. We also notice that the BLEU significantly increases with the number of parameters, while BertScore remains relatively unaffected by the model. Therefore, all models achieve outstanding results in terms of BertScore, thus proving that the generated text is semantically correct. However, models with a large number of parameters tend to produce more long n-gram sequences matching the spell description in the test set causing overfitting.

[10] https://github.com/huggingface/evaluate

4.2. Qualitative Evaluation

For qualitative evaluation, we set up an experiment involving *D&D* players. To facilitate this investigation, we developed a Telegram chatbot, which was the primary interface for engaging with participants. This allowed us to interview players and easily keep track of their answers. The chatbot is designed to show the player one spell at a time, with 10 spells for the whole session. Additionally, players retain the freedom to interrupt the questionnaire at any point and resume their progress later, as their responses are continuously saved throughout the interaction process. Users could also decide to perform multiple experimental sessions, in which case we ensured that no spell seen by a user in a previous experiment could be seen again in a new one.

For each spell, following the questionnaire proposed in (Newman and Liu, 2022), there are three questions asked to the player. The questions, along with the kind of expected answers, are shown in Table 5.

For the qualitative evaluation, we consider the five best-performing models: `opt-2.7b`, `opt-6.7b`, `opt-13b`, `llama2-7b` and `llama2-13b`. During each session, we present the user with ten spells: one from each model and five written by humans. The spells are randomly selected, which ensures that the user is equally likely to encounter a human-written spell and a machine-generated one. For the generation, we employed Top-p Sampling (also known as "Nucleus Sampling") (Holtzman et al., 2019) with $p = 0.9$ (a commonly used value, in particular, (DeLucia et al., 2020) study that a value in the range $[0.7, 0.9]$ is best in narrative generation). During the experiment via the Telegram bot, the users are not informed of the total number of AI-generated and human-written spells. This omission is aimed at maintaining an unbiased environment throughout the whole experiment.

A total of 13 users completed at least one experiment session for a total of 16 sessions. The overall results can be seen in Table 4, while the results per model can be seen in Table 6.

Table 4 shows that 73% of AI-generated spells are identified by the users, but at the same time, 35% of human-written spells are recognized as written by the AI. Furthermore, the table also provides

Model	BLEU	BertScore (Precision)	BertScore (Recall)	BertScore (F1)
gpt2	0.093	0.835	0.851	0.842
gpt2-medium	0.121	0.858	0.855	0.856
gpt2-large	0.110	0.861	0.857	0.858
gpt2-xl	0.149	0.862	0.856	0.858
opt-125m	0.082	0.850	0.845	0.846
opt-350m	0.113	0.860	0.852	0.856
opt-1.3b	0.090	0.861	0.854	0.857
opt-2.7b	0.123	0.870	0.862	0.866
opt-6.7b	0.097	0.867	0.860	0.863
opt-13b	0.135	0.870	0.864	0.867
llama2-7b	0.175	0.877	0.874	0.875
llama2-13b	**0.188**	**0.880**	**0.876**	**0.877**

Table 3: Results of the quantitative evaluation.

	Human	*AI*
Correctly Identified	66%	71%
Average Rule Conformity	3.75 (4.20)	2.88 (2.50)
Average Playability	3.49 (3.87)	2.76 (2.33)

Table 4: Overall results of the qualitative evaluation. The Human column refers to human-written spells, while the other one to AI-generated spells.

QUESTION 1 What do you think made this? **ANSWER 1** - Human - AI - I have already seen this spell, I know it was written by a human
QUESTION 2 How well do you think this spell conforms to *D&D*'s rules? **ANSWER 2** 5-point Likert scale (1 = "Doesn't fit with the rules at all", 5 = "Would fit in right alongside official spells")
QUESTION 3 Would you play/allow this spell? **ANSWER 3** 5-point Likert scale (1 = "Definitely wouldn't", 5 = "Definitely would")

Table 5: Questionnaire of the Telegram Chatbot. The user is asked to answer all three questions for each spell.

the results (in brackets) computed only on the instances correctly classified as human-written or AI-generated by users. These results show that when spells written by humans are correctly classified by users, the "Average Rule Conformity" and "Average Playability" both increase. In contrast, in the case of AI-generated spells, the two scores slightly decrease.

Analyzing the qualitative results for each model, we obtain that the best models are `opt-2.7b` and `llama2-13b`, while the worst model is `opt-6.7b` as a lower value of "Correctly Identified" means that humans are not able to distinguish AI written spells from human ones. These results confirm the quantitative evaluation where the `opt-2.7b` overcomes `opt-6.7b`. Also the average rule conformity follows the behaviour of the correctness. Furthermore, the "Average Playability" confirms that `opt-6.7b` is the worst performing model, while the other models obtain similar results. Considering the results in brackets, only `opt-2.7b` provides a satisfying "Average Rule Conformity" score, while other systems are under the average.

Table 7 to Table 11 show examples of spells generated by each of the best models, which were also the subject of the qualitative evaluation.

5. Conclusions

In this paper, we presented our exploratory study on using LLMs for a Natural Language Generation task within the domain of tabletop games. More specifically, we focus on *Dungeons & Dragons*, one of the most popular tabletop games in the world, which was published 50 years ago. *D&D* is a complex game that allows players to create their own worlds and characters.

For this work, we decided to focus our attention on a specific aspect of the game: spells. Spells are very interesting since they can affect single characters or fiends but also completely change the outcomes of a story.

We took into account a total number of 12 models, subdivided into three main families: `gpt2`, `opt`, and `LLaMA2` and evaluated their performance both

	opt-2.7b	*opt-6.7b*	*opt-13b*	*llama-2-7b*	*llama-2-13b*
Correctly Identified	63%	75%	75%	75%	69%
Average Rule Conformity	3.13 (3.1)	2.94 (2.67)	2.56 (2.25)	2.5 (2.00)	3.25 (2.64)
Average Playability	2.81 (2.4)	2.63 (2.17)	2.56 (2.17)	2.88 (2.58)	2.88 (2.36)

Table 6: Per-model results of the qualitative evaluation

Name: Disintegrate
Level: 2
School: Abjuration
Classes: Sorcerer, Wizard
Casting time: 1 Action
Range: 60 Feet
Duration: Instantaneous
Components: S
Description: You disintegrate a creature of your choice within range, causing it to disintegrate into fine mist. Each creature in a 10-foot cube centered on the target must succeed on a Dexterity saving throw or take 3d6 force damage and be stunned until the end of your next turn.
If you target a creature that you can see and have line of sight with, they do not have to make the saving throw.

Table 7: Example of spell generated by the `opt-2.7b` model.

Name: Sunfire Blow
Level: 1
School: Evocation
Classes: Sorcerer, Wizard
Casting time: 1 Action
Range: Self (30-Foot Cone)
Duration: Instantaneous
Components: S
Description: You hurl a magical punch, causing a pillar of fiery light to erupt from your hand. Each creature in a 30-foot cone must make a Constitution saving throw. A target takes 8d6 fire damage on a failed save, or half as much on a successful one. If a creature starts its turn in the area of the cone, or ends its turn in the area of the cone, it must make the saving throw with disadvantage.
At Higher Levels. When you cast this spell using a spell slot of 2nd level or higher, the damage increases by 1d6 for each slot level above 1st.

Table 8: Example of spell generated by the `opt-6.7b` model.

Name: Wall of Force
Level: 0
School: Evocation
Classes: Wizard
Casting time: 1 Action
Range: 60 Feet
Duration: Concentration, up to 1 hour
Components: V, S
Description: You and up to five other creatures of your choice are surrounded by magical force, up to 1 foot thick. Creatures inside the wall can only be attacked by spells or creatures that can cross the wall (such as by teleportation or a nonmagical teleporter).
While the wall is in effect, any weapon attack is made at advantage, and the target automatically passes the Dexterity saving throw it takes to avoid taking damage from said attack. Creatures also have advantage on the Strength check it takes to damage this wall.

Table 9: Example of spell generated by the `opt-13b` model.

Name: Healing Mist
Level: 3
School: Necromancy
Classes: Bard, Cleric, Paladin
Casting time: 1 Action
Range: 150 Feet
Duration: Concentration, up to 1 minute
Components: V, S
Description: A mist infused with divine magic envelops a creature of your choice that you can see within range. The target regains 3d6 hit points upon the mist being cast, and also regains 1d6 hit points each round that you maintain concentration. Upon the spell ending, the target takes 2d6 necrotic damage.
At Higher Levels. When you cast this spell using a spell slot of 4th level or higher, the hit points regained by the target increases by 1d6 for every level above 3rd.

Table 10: Example of spell generated by the `llama-2 7b` model.

in a quantitative and qualitative evaluation.

For the quantitative evaluation, we compute the BLEU and the BertScore. From the analysis of the results, we can confirm that the number of parameters positively affects the results obtained by the models, especially in terms of the BLEU score. Next, we took into account the best five performing models to conduct an in-vivo evaluation with actual *D&D* players. The results of this qualitative evaluation confirm the results of the quantitative evaluation. Finally, we also publicly available all the models, datasets, and results of this work, believing that this will foster further research on this topic.

In future works, we plan to extend the use of LLMs to other aspects of the game, such as dialogue continuation or the simulation of other game mechanics. We plan to enhance the capabilities of our models by refining them for additional tasks through fine-tuning. Moreover, we will replicate the pipeline to other open-source models and improve

Name: Shield of Sand
Level: 3
School: Transmutation
Classes: Druid, Sorcerer, Warlock, Wizard
Casting time: 1 Action
Range: Touch
Duration: Concentration, Up To 1 Hour
Components: V, S, M
Material cost: a handful of sand
Description: You create a shield of swirling sand in the palm of your hand. The shield is a dull gray in color, and has an AC of 15 and 15 hit points. The shield disappears if you let go of it. If the shield is reduced to 0 hit points, it disintegrates and the creature must make a Dexterity saving throw. On a failed save, the creature takes 5d6 bludgeoning damage and is blinded until the start of its next turn.

At Higher Levels. When you cast this spell using a spell slot of 4th level or higher, the spell's duration increases to 8 hours. When you cast this spell using a spell slot of 6th level or higher, the spell's duration increases to 24 hours. When you cast this spell using a spell slot of 8th level or higher, the spell's duration increases to 1 week

Table 11: Example of spell generated by the `llama-2 13b` model.

the qualitative evaluation by involving more users.

Acknowledgment

We acknowledge the support of the PNRR project FAIR - Future AI Research (PE00000013), Spoke 6 - Symbiotic AI (CUP H97G22000210007) under the NRRP MUR program funded by the NextGenerationEU.

6. Bibliographical References

Chris Callison-Burch, Gaurav Singh Tomar, Lara J Martin, Daphne Ippolito, Suma Bailis, and David Reitter. 2022. Dungeons and dragons as a dialog challenge for artificial intelligence. *arXiv preprint arXiv:2210.07109*.

Alexandra DeLucia, Aaron Mueller, Xiang Lisa Li, and João Sedoc. 2020. Decoding methods for neural narrative generation. *arXiv preprint arXiv:2010.07375*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Pax Newman and Yudong Liu. 2022. Generating descriptive and rules-adhering spells for dungeons & dragons fifth edition. In *Proceedings of the 9th Workshop on Games and Natural Language Processing within the 13th Language Resources and Evaluation Conference*, pages 54–60, Marseille, France. European Language Resources Association.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. pages 311–318.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Revanth Rameshkumar and Peter Bailey. 2020. Storytelling with dialogue: A critical role dungeons and dragons dataset. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5121–5134.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Andrew Zhu, Karmanya Aggarwal, Alexander Feng, Lara J Martin, and Chris Callison-Burch. 2023. Fireball: A dataset of dungeons and dragons actual-play with structured game state information. *arXiv preprint arXiv:2305.01528*.