Using Language Models to Convert Between Natural Language and Game Commands

Stefan Papazov and **Wesley Gill** and **Marta Garcia Ferreiro**
Andrew Zhu and **Lara J. Martin** and **Chris Callison-Burch**
University of Pennsylvania
School of Engineering and Applied Science
{spapazov, wesgill, martagf, laramar, ccb}@seas.upenn.edu

Abstract

Dungeons and Dragons is a popular tabletop role-playing game that has been adapted to online play. In this paper, we look at enhancing a Discord Bot called Avrae that is developed by D&D Beyond to help with online play. Avrae enables users to manage gameplay through Unix-like commands. We explore using language models to automatically translate player dialogue into Avrae's commands. We use GPT-3's few shot learning and fine tuning capabilities and achieve 64% accuracy. We also explore the reverse direction, where commands are rendered as descriptive text, suggesting that it may eventually be possible to combine Avrae and LMs to create a system that is capable of role playing alongside players.

1 Introduction

Dungeons and Dragons (Gygax and Arneson, 1974) is a popular roleplaying game that has experienced a revival in part thanks to the introduction of online game play. D&D players create fantasy characters and embark on an adventure guided by another person given the role of Dungeon Master (DM). During their adventure, they may engage in combat, complete quests, and solve challenges while engaging in imaginative "role-play" – acting out or describing what their characters would do in a situation. Players are given the freedom to tell the story as they go, with each player deciding the actions taken by their character at each turn. The final outcomes are determined by player dice rolls and the results of the action are then described by the DM. D&D involves managing a large number of rules (Wizards of the Coast, 2018). These include rules that govern combat, that track game state elements on character sheets, and that determine whether dice rolls are successful or not.

Avrae[1] is a Discord bot that allows players and DMs to perform dice rolls. Avrae allows players to type in a command to roll their attacks, ability checks, and saving throws. Avrae interfaces with their character sheet to add appropriate modifiers to the dice rolls. It is programmed with many of the rules and monster stat blocks from D&D so it is able to automatically determine whether a roll was successful. This helps to simplify some of the rule-based aspects of D&D gameplay, like tracking combat. The mechanism by which players and DMs interact with Avrae by messaging the bot in Discord, sending it commands that it then interprets. These commands are similar in style to Unix commands, and have the same disadvantages of having strict syntax, and being somewhat obscure and difficult to learn.

Natural Language	Avrae Command
I shoot my bow at the goblin	`!attack bow -t go1`
Dirk listens for footsteps	`!check perception`
Strength save	`!save strength`
I cast fire bolt on the fourth zombie	`!cast "fire bolt" -t zo4`

Table 1: Examples of natural language sentences and their corresponding Avrae commands.

In this paper, we examine the task of automatically translating from a sentence written in English onto a corresponding command for Avrae to execute. For example, if a player says "I shoot my bow at the goblin" then the system should transform that onto the Avrae command `!attack bow -t go1`. Table 1 shows additional examples. Our contention is that by reducing the players' need to remember Unix-style commands that they may instead spend more time on the role-playing aspects of the game.

We also examine the reverse direction. Given an Avrae command (and optionally some additional context of the recent conversational history), we

[1] https://avrae.io/

experiment with having our system verbalize the command in English. In this direction, commands are rendered as descriptive text. Our pilot study suggests that it may eventually be possible to combine Avrae with large language models to create a system that is capable of role playing alongside players, or that can be used to produce flavor text that they players can use.

In this paper, we first establish a baseline for translating English into Avrae commands based on work for translating natural language to Bash (Fu et al., 2021). Second, we using GPT-3 for the task, using few shot learning, fine-tuning. We also try decomposing the task into multiple stages to simplify the command translation task. Finally, we experiment with fine tuning GPT-3 to perform the task of generating descriptive D&D text given Avrae contexts as input.

2 Related Work

Previous NLP research has examined translating natural language into commands. For example Lin et al. (2018a) focused on English-to-Unix commands and assembled a dataset **NLC2CMD** which maps 9,000 English-command pairs, covering over 100 unique Bash utilities. A performance baseline of 36 % accuracy was established by training seq-to-seq (Sutskever et al., 2014) and CopyNet (Gu et al., 2016) models, which were known for state of the art performance in classical natural language translation tasks (Lin et al., 2018b). IBM subsequently hosted a competition aimed at achieving state of the art translation performance over the **NLC2CMD** dataset (Agarwal et al., 2021). The winning team, Magnum, achieved an accuracy of 52% using an ensemble of five separately trained Transformers with Beam Search enabled. They decreased vocabulary size by masking certain terms in a command and replacing them with general tokens. This increased prediction accuracy, but was only effective for utilities in the command which had a bounded number of possible inputs (e.g. the start of a bash commands like *ls, cd, mkdir, etc*) (Fu et al., 2021). A fine tuned version GPT-2 achieved the second highest performance among all models 51%. Baseline performance was evaluated through TF-IDF information retrieval and achieved an accuracy of 36% (Agarwal et al., 2021). We note that accuracy in this considered only command structure correctness, and did not account for correct parameter inputs into flags (Agarwal et al., 2021).

Lin et al. create their own natural language to command dataset by web scraping 5000 instances bash commands and their descriptions. They train an RNN based model, Talina, which achieves a 30% top-1 accuracy at predicting commands (Lin et al., 2017). The Talina model was also tested on the **NLC2CMD** dataset and achieved a slightly lower performance of 27% top-1 accuracy (Lin et al., 2018b). Overall, the research indicates that transformers are better suited to natural language to command translation than LSTM, RNN or GRU based architectures (Agarwal et al., 2021; Fu et al., 2021)

Our second task, on generating descriptive text from commands is similar to previous work done in fictional story generation. Fan et al. (2018) show that providing language models with an input prompt which alludes to the overall story plot, can improve the quality of story generation, as compared to alternatives. In our study we hypothesize that Avrae commands written over a D&D campaign, would form a similar plot line schema. Goldfarb-Tarrant et al. (2020) introduce a system for formulating plot structures as ordered events, and later use these to generate more relevant and cohesive stories. Similarly Yao et al. (2019) employ hierarchical story generation to interchangeably predict the story plot and story text, thereby improving story line diversity and cohesion as measured by numerical and human evaluation metrics. We posit that Avrae commands could prove helpful in story reconstruction by acting a schema for D&D adventures.

Our dataset annotating D&D play-by-post games with Avrae commands adds to an expanding set of NLP data related text generation for D&D and text adventure games like TextWorld (Côté et al., 2018), Role-Playing Game Transcripts (Louis and Sutton, 2018), LIGHT (Urbanek et al., 2019), STORIUM (Akoury et al., 2020), and a D&D dataset based on the Critical Role podcast (Rameshkumar and Bailey, 2020).

3 Annotated Data

In order to apply NLP methods to learn English-to-command mappings, we needed a source of annotated data. For this study, we manually annotated a set of game transcripts with their corresponding Avrae commands. The game transcripts were collected by Anonymous (in submission) from a Play-By-Post forum on D&D Beyond that did not use

Command	Frequency	Number of Turns
check	357	286
attack	108	93
cast	57	51
save	9	9
other	7	7
total	538	446

Table 2: Counts of Hand Labeled Data. **Frequency** measures the total number of times a command type was used across all turns and **Number of Turns**, is the number of turns which had each command type.

Avrae. The dataset contains several attributes regarding the game state and actions at each turn. We analyzed a single D&D campaign from the Play-By-Post forum, which had 4,289 conversational turns of which 1,739 are fight sequences and 1,949 contain some type of character action (attack or ability check).

Without augmentation, the Play-By-Post dataset is insufficient for our translation task since it includes no Avrae Commands. So, we hand labeled a sample of 446 turns from the Play-By-Post dataset with their associated Avrae commands. 98% of these hand-labeled turns were composed of four commands–check, save, attack, and cast–despite Avrae offering a much wider range of commands. One reason for this is that many Avrae commands pertain to functionality such as creating character sheets, adding players to a server, looking up rules, or looking up item information. Text corresponding would fall outside of what typically happens in D&D Beyond Play-By-Post data which. We thus chose to limit this study's focus to commands that track initiatives like player combat, exploration and saves. Table 2 gives information about the distribution of annotations in our dataset.

3.1 Challenges and Edge Cases

A subproblem of our English-to-command task is determining if an instance of player dialogue contains an Avrae command. Only 45% of all turns in the Play-By-Post dataset contain player actions that could potentially translate to commands. Previous studies translating natural language to bash commands never considered this scenario (Fu et al., 2021; Agarwal et al., 2021; Lin et al., 2017, 2018b). In our study, we therefore choose to discount these instances and only focus on data points which have associated commands.

Another divergence in our dataset is that player dialogue often maps to more than one command. For example a player may want to carry out a check and subsequently attack an enemy or vice versa - this is represented as two separate Avrae commands.

> **Text**: *"I walk up to the mouth of the cave and peer inside. Sneakily 15 Sneak 25 Perception"*
> **Commands**: `!check stealth`; `!check perception`

4 Natural Language to Commands

In this section we focus on translating player dialogue into Avrae commands. We define several evaluation metrics, establish a baseline and run a series of experiments using Open AI's GPT-3 davinci and curie models.

4.1 Evaluation

4.1.1 Ordered Translation Accuracy

The first evaluation metric that we consider is ordered translation. In this case, we label a command as correct if the predicted command exactly matches the gold label (ignoring trailing whitespace). We compute overall string match accuracy with P the prediction, and L the label as:

$$OTA = \frac{\sum_{i=1}^{N} 1\{P_i = L_i\}}{N}$$

Recall that player dialogue could induce multiple Avrae commands, and the order in which these commands are issued may not necessarily be important. Ordered translation accuracy underestimates the true efficacy of the system in these cases.

4.1.2 Unordered Translation Accuracy

We define unordered translation accuracy to account for different orderings in the predicted commands. For a given data instance, consider its label L' to be the set of gold string commands and its prediction P' to be the set of string command predictions, parsed by splitting on semicolons. We define unordered accuracy as:

$$UTA = \frac{1}{N} \sum_{i=1}^{N} \frac{|P_i' \cap L_i'|}{|L_i'|}$$

4.1.3 Translation Precision

In addition to accuracy, we also measure the precision of our translations in terms of the number of relevant commands (unordered) that we predict.

For a given data instance, consider its label L' to be the set of gold string commands and its prediction P' to be the set of string command predictions. We define translation precision as:

$$TP = \frac{1}{N} \sum_{i=1}^{N} \frac{|P'_i \cap L'_i|}{|P'_i|}$$

4.2 Baseline Performance

No previous work has been done to translate natural language into Avrae commands, making it hard to directly compare our performance to previous literature. To obtain a baseline, we considered the best performing model in the IBM **NLC2CMD** competition, an ensemble of five separately trained Transformers with Beam Search, and re-trained it on our Play-By-Post data. Our only modifications were that we reduced the number of training steps to 600 and trained on a single GPU, whereas the original paper trains for 2000 steps on two GPUs (Fu et al., 2021). This model achieved a baseline performance of 43.6% unordered translation accuracy and 35.0% ordered translation accuracy.

4.3 Prompt Format

A key hurdle that separates Avrae command translation from previous work is that player dialogue does not always provide the entire context of the game state. All players know the game state information and need not repeat it at every turn for others to understand their actions. Consider the following example:

> **Text**: *"I look at these creatures and I cast ... as I call out to the heavens....MAGIC MISSILE"*
> **Command**: `!cast 'magic missile' -t go1 -t go2`

The player refers to the goblins in the current combat sequence as "these creatures", making it difficult even human readers to infer the targets of his spell casts without further context. To handle this we pre-process our prompts to include certain aspects of game state including the fantasy race and class of the player currently speaking,as well as the active monsters in game. With this, we format model inputs as follows:

> **Race**: human | **Class**: wizard | **Monsters**: goblin 1, goblin 2 | **Speech**: I look at these creatures and I cast ... as I call out to the heavens....MAGIC MISSILE"
> **Command**:
> `!cast 'magic missile' -t go1 -t go2`

4.4 Few Shot Learning

Previous approaches to translating natural language to commands rely entirely on training and weight updates. Large language models have been shown to be useful for many NLP tasks, so we experimented with using GPT-3's fine tuning and few shot learning capabilities (Brown et al., 2020). In we consider several approaches to constructing few-shot prompts and evaluate their performance. We algorithmically created our prompts in order to decrease the dependence on needing good human-written prompts.

4.5 Fine Tuning

In addition to few shot learning, we finetune Davinci, OpenAI's largest GPT-3 model (Brown et al., 2020). Previous work has shown that fine tuning GPT-3 shows improved performance in zero shot and few shot settings (Wei et al., 2021). For the remainder of this study we refer to these experiments as *GPT-3 Fine Tune*. We use 308 training examples for fine tuning. We also carry out two further experiments with these fine tuned models

4.5.1 Task Decomposition

We experiment with breaking down command translation into three separate sub problems:

1. Classify the type of command (cast, attack, save, or check)
2. If the command is a check or save, identify the type (history, athletics, etc.)
3. If the command is an attack or cast translate based on few shot examples provided by identifying the attack/spell type and the target

4.6 Natural Language to Command Results

Table 3 summarizes the performance of our models on the Natural Language to Commands task. We use a total of 103 Play-By-Post turns as testing data selected at random. We find that our decomposition model performs best overall, with 64% unordered translation accuracy and 40% ordered translation accuracy. A

For translation precision, the task decomposition model show a lower overall precision (43%)

Model	Ordered Accuracy	Unordered Accuracy	Overall Precision	Attack Precision	Cast Precision	Check Precision
NLC2CMD Baseline	0.35	0.44	0.48	0.11	0.00	0.67
GPT-3 Few Shot	0.32	0.57	**0.53**	0.29	0.23	0.69
GPT-3 Fine Tuned	0.27	0.60	0.51	0.11	0.14	**0.73**
GPT-3 Task Decomposition	**0.40**	**0.64**	0.43	**0.34**	**0.27**	0.49

Table 3: The performance of the baseline model of our GPT-3 variants on our English-to-Command Task

as compared to the GPT-3 few shot model (53%), GPT-3 fine tuned and NLC2CMD baseline (48%). However, we also highlight that task decomposition setups achieve relatively higher precision for attack and cast commands compared to alternative approaches. We posit that this comes as a virtue of first identifying the command type (attack, cast, check, etc.), before generating a translation.

Qualitative examples of the models' outputs are given in Appendix A

5 Commands to Natural Language

In addition to natural language to command translation, we believe that the reverse direction also offers a rich area of study: using Avrae commands to generate descriptive text. We investigated if Avrae commands can be used as input to large language models to generate text as an aid for Dungeons and Dragons storytelling.

A signature trait of D&D campaigns is that, after each player has taken their turn, the DM will summarize that overall result of the character's actions, and further build upon the story line by introducing new actions from non player characters (NPCs). This response provides a rich summary of the events that have taken place in the previous series of turns.

Given that Avrae commands issued by players are essentially a schema for the events that occurred in a given turn sequence, we are curious to see if these could be leveraged by Language Models to generate "realistic" DM summaries.

5.1 Data Pre Processing

Given the structure of a D&D adventure, we are able to break our data into series of rounds consisting of two parts: (1) player turns and (2) DM summary of the events that occurred after each player outlined their actions. In any D&D adventure, the rules dictate that players take turns outlining the actions taken by their characters and roll a dice to determine if their outcome is a success. Subsequently, the DM will summarize the events that occurred in the form of a short description based on the action of each player the values of their dice rolls.

We define a round to be a sequences of turns, in which the players state their actions, and the DM summarizes the events that occurred. Using this definition we extract 15 turn sequences with hand labeled Avrae commands from our dataset. Our process for extracting turn sequences is as follows:

1. Segment the entire Play-By-Post data set into turn turn sequences which start with a player turn and end with a DM turn
2. Iterate through each turn sequence. If the turn sequence contains player turns which all have associated Avrae command labels, then store this turn sequence as part of our dataset
3. Discard all remaining turn sequences.

We use these turn sequences to generate DM summaries of player turns through fine tuning.

5.2 Description Generation Models

We fine tune four different GPT-3 models to generate DM summaries based on different levels of input game context. These models differ only in the prompts we provide to them. Our prompts denote the relevant state of the previous player round and the completion is the DM's English-prose summary in the real campaign.

Prompts are compose of three types of information:

- Character details (character name, fantasy race and class)
- Avrae commands corresponding to a player's turn (manually annotated)
- Text of the turn.

The race and class of a character also play important roles in determining which actions are valid.

	Human	Avrae Only	Avrae + Context	Avrae + Dialogue	Avrae + Both
Cohesion	3.9	3.9	3.9	4.1	**4.1**
Interest	**4.4**	4.1	4.0	4.3	4.3
Relevance	4.2	4.1	4.0	4.1	**4.2**

Table 4: Human Evaluation Likert Scores for our Description Generation Task (mapping from Game Commands to Natural Language Sentences)

We experiment with variants on the prompt to test the ability of a fine-tuned GPT-3 model to generate a DM's descriptions of what happened given:

1. Only the player's Avrae command
2. The player's Avrae command, and several prior turns for context.
3. The player's Avrae command, and the player's dialogue for their turn.
4. The player's Avrae command, both context and player's dialogue.

5.3 Evaluation

We use three criteria to evaluate the quality of a DM summary - cohesion (whether the output is logical), interestingness (how engaging the output is for a reader as a fictional story), and relevance (how closely the output pertains the input commands). We hired human evaluators on Mechanical Turk to judge the quality of our generated summaries with respect to the three aforementioned benchmarks on a 1-5 Likert scale. We design one Mechanical Turk task to collect judgments for all three criteria simultaneously. Our MTurk task design is given in Appendix C.

5.4 Evaluation Dataset

When cleaning our data, we found that many rounds of gameplay had a majority of turns with no associated Avrae commands. To resolve this, we first filtered the data to include only those rounds in which at least 80% of the turns had an Avrae commands. This left us with 106 rounds of gameplay. We then split the data with an 80/20 train-test split in which the first 80% of the rounds (chronologically, i.e. those at the start of the campaign) were used to train the model and the final 20% of the rounds were reserved for testing. This left us with 84 instances in our training set and 22 instances in our hold out set. With 5 assignments per HIT and 22 instances used for evaluation, each model was scored a total of 110 times on each of the three categories.

5.5 Description Generation Results

Example outputs are given in Appendix B. Table 4 summarizes the average scores on each of the three metrics for the four fine-tuned models and the human baseline. Overall, each of the models scored roughly a 4 out of 5 on each of the three categories with all scores in the narrow range from 3.8 to 4.4. This was surprising to us because we thought there would be greater variation in the evaluations of the models, and that the human generations would receive the highest scores on all metrics.

As we hypothesized, adding player dialogue alone helped in every category. Adding context alone helped only in the cohesion category. However, including both context and dialogue improved the model across every category. This may indicate that a combination of structured (Avrae commands) and unstructured (player dialogue) context may help in DM generation or story generation more broadly.

Unsurprisingly, MTurk respondents rated the original human generations with an interestingness of 4.4 while the model that included the Avrae commands, context from the last round, and player dialogue was ranked the most cohesive and relevant, scoring a 4.1 and 4.2 on these two categories, respectively. Moreover, it scores only marginally behind the human topline for interestingness.

6 Discussion

Our study explored the capacity of GPT-3 to translate between natural language and game commands. We find that, in the dialogue to command translation setting, few shot learning can often outperform fine tuning and model training, especially when tasks are decomposed into simpler ones. We caution however, that this better performance may come at the expense of lower translation precision, where the model is “guessing more commands".

We also found that task decomposition setups experience significantly less drop off in unordered translation performance as compared to fine tuning and single prompt approaches, when the prompt length is decreased.

Secondly, we show preliminary findings which suggest that GPT-3 has the potential to generate human-like D&D descriptive dialogue. Our experiment further suggests that, when Avrae command data is provided as a input prompt, GPT-3 tends to generate cohesive and relevant stories as rated by MTurk workers. Further work would be needed to explore if this would also hold true under the scrutiny of expert evaluators.

7 Limitations

Our annotated dataset has several limitations. First, 500 annotated turns is relatively small and fine-tuned models may benefit from larger numbers of annotations. Second, turns were manually annotated post-hoc. It would be better to instrument Avrae to capture commands and log the previous conversation. We have teamed with the Avrae developer and have begun to collect such data. This will also contain more information about the game state such as character sheet data, HIT points, and monsters in combat.

Our models for translating natural language to Avrae commands do not currently check whether the command is a valid command in the Avrae system. Neither our models nor our evaluation takes this into account.

Our human evaluation results indicate that our final model is surprisingly good as a human in generating D&D text according to our MTurk evaluators. Another interpretation of our results points to a potential shortcoming in our methodology: the MTurk ratings might not be that reliable. To have our task completed cheaply and quickly, we did not require that raters have played or DMed a D&D campaign before—they may not even have known what D&D was. For future work, it may be beneficial to commit a greater number of resources to ensure that high-quality ratings are collected from domain experts.

References

Mayank Agarwal, Tathagata Chakraborti, Quchen Fu, David Gros, Xi Victoria Lin, Jaron Maene, Kartik Talamadupula, Zhongwei Teng, and Jules White. 2021. Neurips 2020 NLC2CMD competition: Translating natural language to bash commands. *CoRR*, abs/2103.02523.

Nader Akoury, Shufan Wang, Josh Whiting, Stephen Hood, Nanyun Peng, and Mohit Iyyer. 2020. STORIUM: A Dataset and Evaluation Platform for Machine-in-the-Loop Story Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6470–6484, Online. Association for Computational Linguistics.

Anonymous. in submission. Dungeons and Dragons as a dialog challenge for artificial intelligence.

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. 2020. Language models are few-shot learners.

Marc-Alexandre Côté, Akos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. 2018. Textworld: A learning environment for text-based games. In *Workshop on Computer Games*, pages 41–75. Springer.

Angela Fan, Mike Lewis, and Yann N. Dauphin. 2018. Hierarchical neural story generation. *CoRR*, abs/1805.04833.

Quchen Fu, Zhongwei Teng, Jules White, and Douglas C. Schmidt. 2021. A transformer-based approach for translating natural language to bash commands. *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1241–1244.

Seraphina Goldfarb-Tarrant, Tuhin Chakrabarty, Ralph Weischedel, and Nanyun Peng. 2020. Content planning for neural story generation with aristotelian rescoring. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4319–4338, Online. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *CoRR*, abs/1603.06393.

Gary Gygax and Dave Arneson. 1974. Dungeons Dragons.

Xi Victoria Lin, Chenglong Wang, Deric Pang, Kevin Vu, Luke Zettlemoyer, and Michael D. Ernst. 2017. Program synthesis from natural language using recurrent neural networks. Technical Report UW-CSE-17-03-01, University of Washington Department of Computer Science and Engineering, Seattle, WA, USA.

Xi Victoria Lin, Chenglong Wang, Luke Zettlemoyer, and Michael D. Ernst. 2018a. Nl2bash: A corpus and semantic parser for natural language interface to the linux operating system. *CoRR*, abs/1802.08979.

Xi Victoria Lin, Chenglong Wang, Luke Zettlemoyer, and Michael D. Ernst. 2018b. Nl2bash: A corpus and semantic parser for natural language interface to the linux operating system. *CoRR*, abs/1802.08979.

Annie Louis and Charles Sutton. 2018. Deep dungeons and dragons: Learning character-action interactions from role-playing game transcripts. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 708–713, New Orleans, Louisiana. Association for Computational Linguistics.

Revanth Rameshkumar and Peter Bailey. 2020. Storytelling with Dialogue: A Critical Role Dungeons and Dragons Dataset. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 5121–5134.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215.

Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. 2019. Learning to speak and act in a fantasy text adventure game. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 673–683, Hong Kong, China. Association for Computational Linguistics.

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. *CoRR*, abs/2109.01652.

Wizards of the Coast. 2018. Basic rules for dungeons & dragons.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7378–7385.

A Examples Outputs for Task 1 – Natural Language to Commands

Input: I huck a rock at the one behind the bar. 11 dmg. 5

Gold Label : `!cast 'magic stone' -t zombie`
GPT-3 Few Shot : `!attack rock -t zombie`
GPT-3 Fine Tuned : `!attack 'throw rock' -t zombie`
GPT-3 Task Decomposition : `!cast 'magic stones' -t zombie`

Input: I'm going back to the dead goblin near the horses that stabbed me and I'm going to attempt to break his spine backwards so I may cram his head up his ass. Strength: 20

Gold Label : `!check strength`
GPT-3 Few Shot: `!check strength; !attack 'backbreaker' -t goblin`
GPT-3 Fine Tuned : `!attack 'break spine' -t goblin`
GPT-3 Task Decomposition : `!check strength`

Input: Fjolnir shoulders his bow, runs into the fray and stands over the body of Mosk to obstruct their access to Absinthe- drawing Scimitar and Dagger to attack the northern Bugbear- RH Attack 16 Damage 5 Hunters' Mark 2 LH Attack 21 Damage 2 Hunters' Mark 6

Gold Label : `!attack scimitar -t bugbear`
GPT-3 Few Shot: `!attack scimitar -t bugbear;!attack dagger -t bugbear`
GPT-3 Task Decomposition : `!attack scimitar -t bugbear;!attack dagger -t bugbear`
GPT-3 Fine Tuned : `!attack scimitar -t bugbear;!attack dagger`

Input: I would like to yell up the garbage shoot so everyone can hear me alright I think we should keep Klarg alive for now maybe he could be useful but honestly whatever then take a short rest at the mouth of the cave so we aren't attacked right next to the other room and nobody can take the treasure out. History 15

Gold Label : `!check history`
GPT-3 Few Shot: `!check history`
GPT-3 Fine Tuned : `!check history;!rest 'short rest'`
GPT-3 Task Decomposition : `!check history;!game short rest; !yell up garbage shoot`

B Examples Outputs for Task 2 – Commands To Descriptive Text

Below is an example of a full round of gameplay in the form of a prompt and the resulting summaries by both the human DM and the models explored in Section 5. Note that the prompt is displayed with the previous rounds context and the player dialogue, but only some of the generations had access to these details.

Prompt:
DM Context: And so it is! Watch finished, the dawn brings better health and a rested state to everyone. There is no need to scavenge food, there is enough for the rest of the day. (You would not have got any with that roll anyway) Hymir casts a few spells while everyone else breaks down camp and cooks breakfast. Hymir finds nothing when he casts identify on his mechanical spider. No magic detected. The seemingly empty bottle is actually a potion of invisibility

Character: Fjolnir
Race: dwarf
Class: bard
Commands: !init check stealth
Text: Fjolnir chuckles at the building condition, "I guess a coat of paint may not be enough..." he hesitates... "We need to scout around I think, this place will definitely have something crawling around in it." stealth: 12

Character: Az Emberi
Race: elf
Class: wizard
Commands: !init check stealth; !init check perception
Text: Stealth 15 Looking for hostiles perception 11

Character: Krodmad
Race:
Class: wizard
Commands: !init check stealth Text: Krodmad also begins a quiet survey of the area. Stealth: 20

Character: Fjolnir
Race: human
Class: bard
Commands: !init check stealth
Text: Sign says leave guys I whisper with all seriousness. 18 Stealth

Character: Dirk
Race: human
Class: wizard
Commands: !init check survival; !init check perception; !init check investigation; !init check stealth
Text: I would like to see if I can detect any hostiles perception 15 If I'm able to detect any signs of recent foot traffic or activity investigation 7 survival 10 Stealth 16

Character: Absinthe
Race: Goliath
Class: wizard
Commands: !init check stealth
Text: Stealth: 11

True Completion:
You move quietly down the path. The building on the left is somewhat intact. This small house appears to be in better condition than the ruined and dilapidated structures nearby. The doors are reinforced with heavy iron bands, and thick shutters protect the windows. Past this house you can see more ruins. The one on the right is a complete ruin. It looks as if it might once have been a farmhouse. It is now half swallowed by a dense thicket, with trees growing up through its ruined foundations. There are no signs of movement or sounds. However, there is one set of footprints leading into the building on the left.

Avrae Only:
The group makes it through the door and into the hallway without incident. Fjolnir moves to the right and quietly opens a door to the second floor. The room is unremarkable aside from a rotting bed frame, a small table with a pitcher of ale, and a cot. Az Emberi starts to move quietly to the cot when Krodmad says, "No, you stay here," and they both quietly move to the other side of the room. Absinthe moves to the opposite side of the room, Hymir takes the bed, and Fjolnir goes to the door. Secret door?

Avrae + Dialogue:
You silently examine the building. It's a typical

NE standard building with a large 2 story entrance, 2 windows on each side of the building, and a chimney on the NE corner. The walls are covered with graffiti from some long-dead criminal.

Avrae + Context + Dialgoue:
DM Text: Stealth: 15 You see two humanoids standing around a fire talking. They are both carrying bows.

C MTurk Specifications

For each task, we offer workers compensation of $ 0.06 per HIT and require 5 assignments for each HIT. In other words, we will have 5 workers evaluating the quality of each of our generated summaries, including the original gold label DM summaries from the dataset.

In order to ensure that we obtain good evaluations, we require that workers who access our tasks have a HIT approval rate greater than 90%. Additionally, since all of our summaries are in English, we further ask that workers are located in the United States. This location filter also heightens the chance that workers who complete our task have previously encountered D&D , as the game is more popular in western cultures.

We currently do not plan on introduce a control task or incorporating functionality to assign quality ratings to workers. Still, we plan on releasing a test batch of MTurk tasks as soon as possible, and will use this to determine if further quality control is needed.

C.1 Mechanical Turk Hit Design

We designed our Mechanical Turk task, such that each HIT contains a single generated DM summary. We believe this offers the best avenue for crowd workers to objectively rate each text relative to our criteria. We had otherwise contemplated presenting pairs of DM summaries on each HIT, where one is the gold label and the other is the GPT-3 generated output. Ultimately, we decided against this as we believed it may heavily bias workers against generated it outputs, and make final comparison across setups harder to carry out.

Each HIT, consists of three parts - a backstory, current context, and story components. The backstory component displays the DM summary from the previous sequence (when displaying HITs for setups that did not incorporate DM summaries from the previous sequence we mark this field as “Not Provided"). The current context component, displays the turns in the current turn sequence. For each turn we show the character name, race, class, issued Avrae commands and dialogue (if it is included in the setup). Finally, in the story component we present the generated DM summary for each setup. In addition to this, we also have workers judge the gold label summaries as a baseline.

Finally, workers are presented with three selector fields which correspond to our three criteria. Workers can use a drop down to select their label for interestingness, cohesion and relevance to the input context (see Figure 1).

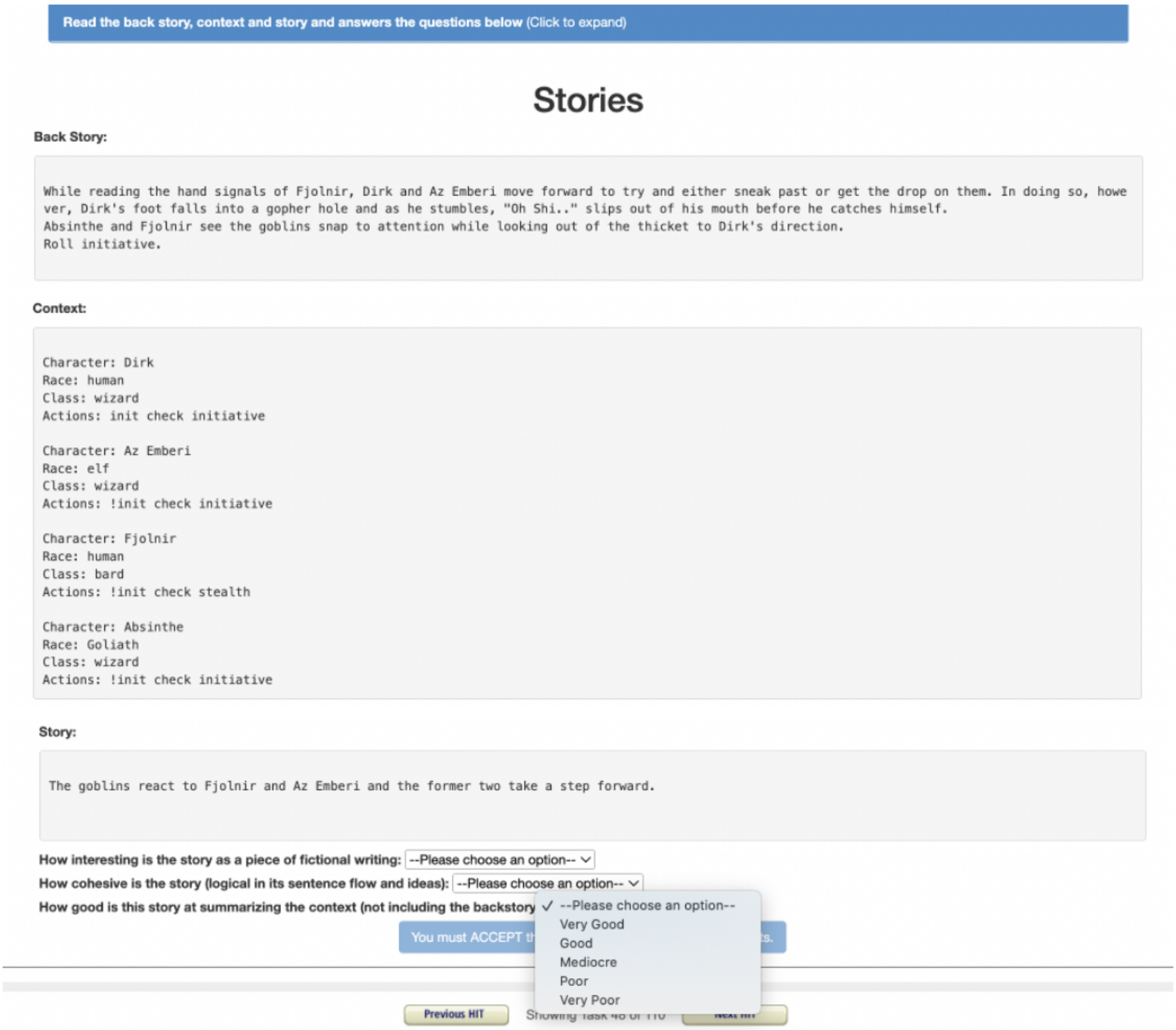

Figure 1: Mechanical Turk HIT Design