CoRRPUS: Code-based Structured Prompting for Neurosymbolic Story Understanding

Yijiang River Dong[1], **Lara J. Martin**[2*], and **Chris Callison-Burch**[1]
[1]University of Pennsylvania
[2]University of Maryland, Baltimore County
riverd@sas.upenn.edu, laramar@umbc.edu, ccb@seas.upenn.edu

Abstract

Story generation and understanding—as with all NLG/NLU tasks—has seen a surge in neurosymbolic work. Researchers have recognized that, while large language models (LLMs) have tremendous utility, they can be augmented with symbolic means to be even better and to make up for many flaws that neural networks have. However, symbolic methods are extremely costly in terms of the amount of time and expertise needed to create them. In this work, we capitalize on state-of-the-art Code-LLMs, such as Codex, to bootstrap the use of symbolic methods for tracking the state of stories and aiding in story understanding. We show that our CoRRPUS system and abstracted prompting procedures can beat current state-of-the-art structured LLM techniques on pre-existing story understanding tasks (bAbI Task 2 and Re^3) with minimal hand engineering. This work highlights the usefulness of code-based symbolic representations for enabling LLMs to better perform story reasoning tasks.

1 Introduction

Stories are a complex form of writing that involve many inter-dependent components, such as causal reasoning (Mostafazadeh et al., 2020; Han et al., 2021), temporal ordering (Basu et al., 2021), social commonsense (Hwang et al., 2021), and the consistent portrayal of facts and events as the story unfolds (Yu et al., 2021; Wilmot and Keller, 2021). Despite the recent uptick in research involving story understanding and reasoning (Qin et al., 2019; Han et al., 2019; Mostafazadeh et al., 2020; Peng et al., 2021; Brahman et al., 2021; Martin, 2021; Brahman, 2022; Chen et al., 2022; Li et al., 2022; Guan et al., 2022; Andrus et al., 2022), there is still a significant gap in the abilities of models to actually understand (or generate) stories at a reasonable level of commonsense. Ranade et al. (2022) provide a recent overview of the area.

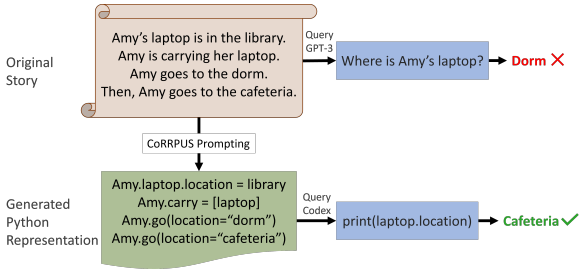

Figure 1: CoRRPUS prompting aids the LLM in properly following the laptop throughout the short story. Meanwhile with natural language prompts, GPT-3 fails to keep track of the laptop through the character Amy's movement. It can follow that Amy has her laptop and that she brought it to the dorm, but it "loses track" of the laptop after Amy goes to the cafeteria.

Although large language models (LLMs) like GPT (Radford et al., 2019; Brown et al., 2020) by themselves can produce text that can be indistinguishable from human-written stories, these models still struggle to generate coherent long-form text (See et al., 2019) & solve simple commonsense reasoning tasks (See Figure 1 for an example.) and therefore, end up only writing at human-level quality less than three-quarters of the time (Ippolito et al., 2020). It is also likely that the majority of this generated text is actually memorized and generated verbatim from human-written stories (Lee et al., 2022).

Over the past few years, researchers have begun to see value in integrating symbolic AI methods—that are consistent and logical—with neural networks—that are flexible and unpredictable (Garcez and Lamb, 2020). These neurosymbolic techniques try to balance the best of both worlds. Nye et al. (2021) draws a parallel between neurosymbolic AI and the System 1/System 2 cognitive science theory (Evans, 2003): neural sequence models are a fast and intuitive system (System 1) that can be improved by adding a logical reasoning system (System 2). Martin (2021)

*Work done while at the University of Pennsylvania.

Findings of the Association for Computational Linguistics: ACL 2023, pages 13152–13168
July 9-14, 2023

and Nye et al. (2021) use GPT-2/3, respectively, to produce candidate story continuations and then use a symbolic state representation to compare against in order to remove suggestions that would be inconsistent.

We extend this work with the use of the Code-LLM Codex (Chen et al., 2021) for structuring and tracking state changes for story understanding. Our system, which we call CoRRPUS (**Co**de **R**epresentations to **R**eason & **P**rompt over for **U**nderstanding in **S**tories)[1], extracts structured information (the story world model) which is then used for reasoning. We compare to neural-only, symbolic-only, and prior works' neurosymbolic systems on two pre-existing story understanding tasks: bAbI and Re^3, which we will describe in Section 4.

Our contributions are as follows:

1. We adapt the Code-LLM model for modeling story worlds and tracking information over time. Our CoRRPUS technique outperforms existing state-of-the-art models on bAbI and Re^3 story understanding tasks.

2. We explore various prompting styles to achieve the best performance from Code-LLMs and report on best practices for working with Code-LLMs.

In the rest of the paper, we will outline a brief history of neurosymbolic work in storytelling and recent use of Code-LLMs. We will then go over CoRRPUS's three prompting methods, followed by a description of the two story understanding tasks bAbI and Re^3 and our experiments with CoRRPUS on both.

2 Related Work

2.1 Neurosymbolic Reasoning in Storytelling

Structured representations have existed in story generation for decades (e.g., (Lebowitz, 1984; Turner and Dyer, 1985)), and while pure symbolic methods are still researched today (Garbe et al., 2019; Christensen et al., 2020; Ware and Siler, 2021), a recent push for combining these structured symbolic methods with probabilistic neural networks has grown. We outline some of these methods here.

Much of the research in story generation uses a hierarchical or multi-stage technique to first plan out the underlying plot (also known as *fabula*) or other underlying structures, and then generate natural language sentences that are informed by the structure (Martin et al., 2018; Fan et al., 2018; Tambwekar et al., 2019; Yao et al., 2019; Goldfarb-Tarrant et al., 2019; Ammanabrolu et al., 2020; Rashkin et al., 2020; Guan et al., 2020; Sun et al., 2022; Yang et al., 2022).

Other popular ways to include structure to enhance story generation include using external resources (Huang et al. (2020) with ConceptNet (Speer et al., 2017), Martin (2021) with VerbNet (Brown et al., 2019), Peng et al. (2021) with ATOMIC (Sap et al., 2019)) or by extracting information from the original CoRRPUS beyond the series of events, such as characters' emotions toward each other (Rao Vijjini et al., 2022) or knowledge triplets (Alabdulkarim et al., 2021). Similarly, summary information of the story so far (Callison-Burch et al., 2022), and a combination of summaries and character relationships (Si et al., 2021) have been used to augment storytelling dialog generation.

Researchers have also seen the benefit of neurosymbolic methods by using external knowledge bases (Zhang et al., 2021) or symbolic representations (Li et al., 2022) to augment a transformer's ability to perform story understanding.

Most relevant to our work, however, is the extraction of a story world or world model. This work uses a dynamic external representation in order to keep track of the story while the system is generating in order to make for more consistent stories. So far this has been done with world state dictionaries (Martin, 2021), object-oriented notation (Nye et al., 2021), and knowledge graphs (Andrus et al., 2022).

2.2 Introduction of Code-LLMs for Reasoning

Recently, thanks to the LLMs and large-scale code training data (Husain et al., 2019), many breakthroughs has been made in automatic code synthesis (Feng et al., 2020; Clement et al., 2020; Chen et al., 2021). Code-LLMs have been shown to be adept at logical reasoning (Wei et al., 2022), numerical reasoning (Cobbe et al., 2021), theorem proving (Wu et al., 2022), and linguistic reasoning tasks, such as the command composition task SCAN (Lake and Baroni, 2018) as shown by Zhou et al. (2023).

Madaan et al. (2022) show that Code-LLMs perform better than LLMs in various structured commonsense reasoning tasks including procedural rea-

[1]Code can be found here: https://github.com/dong-river/CoRRPUS

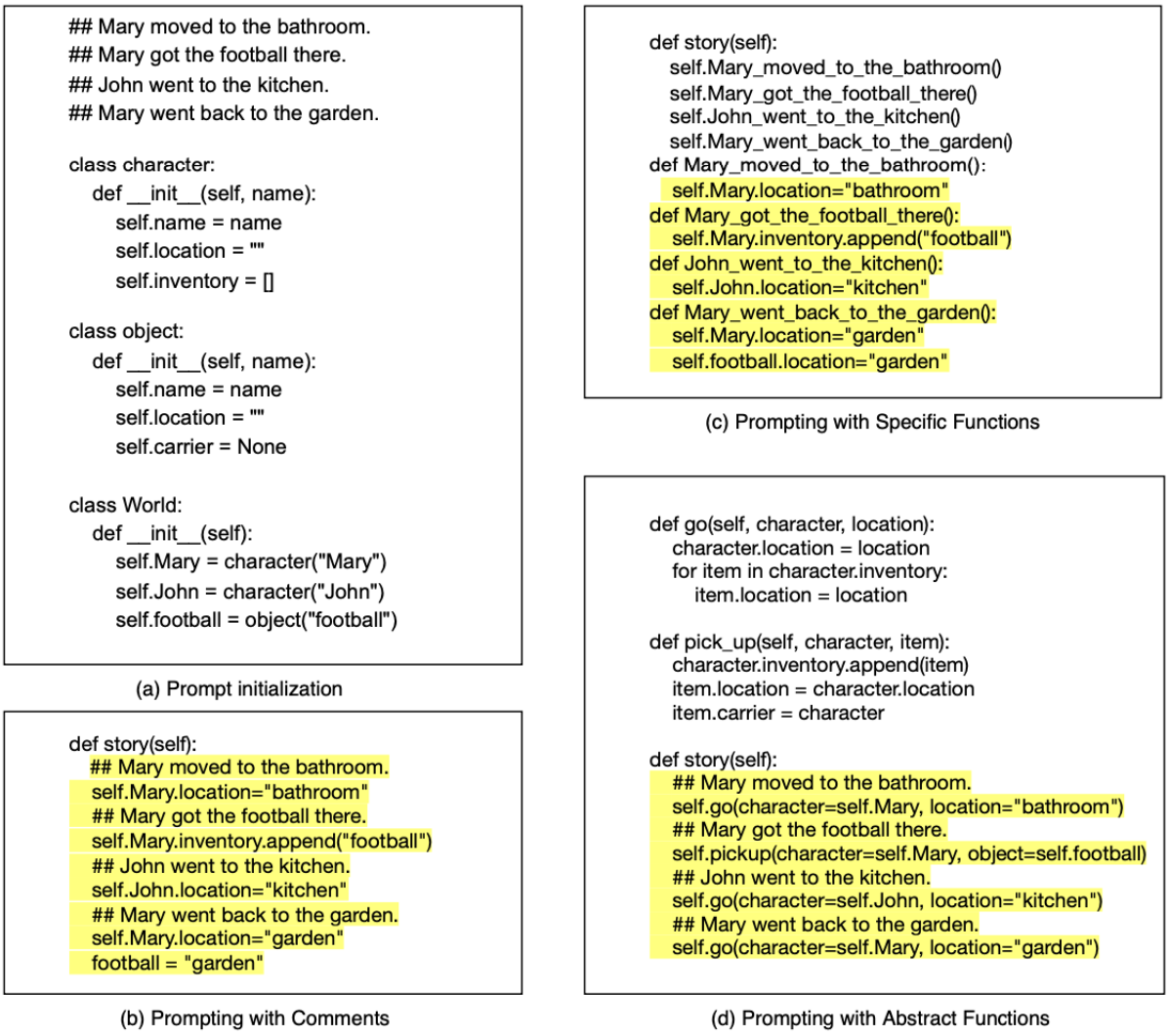

Figure 2: An example prompt used for bAbI Task 2. All three prompting methods have the same prompt initialization (a) followed by their respective additional functions (b, c, or d), found inside the World class (end of a). That is, for the 1-shot example, CoRRPUS would be provided (a) + (b, c, or d) depending on the prompting method. To prompt for the next story, CoRRPUS is given (a) + the non-highlighted of (b, c, or d). The highlighted section would then be generated by CoRRPUS.

soning and entity state tracking, where they used a graph-based representation. Similarly, we prompt Code-LLMs to extract structured world model for neurosymbolic story understanding.

3 The CoRRPUS Prompting System

In this work, we examine the types of prompts for tracking symbolic story state representations using OpenAI's Code-LLM, Codex (Chen et al., 2021). All experiments that use the CoRRPUS prompt system are conducted with code-davinci-002 using OpenAI's API (which was free for research purposes). Following Holtzman et al. (2020), we use nucleus sampling with top-p value equals to 0.95. We set the temperature to 0 when no diverse generation is needed (i.e., answering multiple choice questions from the bAbI task, Section 4.2) and set the temperature to 0.7 when diverse generation is needed (such as in the Re^3 task, Section 4.3). All the experimental results come from a single run.

We provide the Codex model a collection of classes in Python to represent a model of characters and objects that we want to track in the story, in addition to an initialization of the specific characters and objects for the current story, which is presented in a `World` class. See Figure 2a and Appendix 8.1 for examples of these classes. In addition to this information for the current story, we also provide

Codex a 1-shot example of the full process we want it to complete.

Formally, we define the problem as: given a story $\mathcal{S} = [S_0 : S_1 : ... : S_n]$ and a story world state initialization $\mathcal{W}_0$, we want the model to update the story world state $\mathcal{W}_i$ until the story is complete $\mathcal{W}_n$. After each sentence $\mathcal{S}_i$ of the story, the model updates the world state using some black-boxed update function $\mathcal{U}_i$. Thus, the final world state $\mathcal{W}_n$ is obtained via $\mathcal{W}_0 \xrightarrow{\mathcal{U}_0} \mathcal{W}_1 \xrightarrow{\mathcal{U}_1} .... \xrightarrow{\mathcal{U}_{n-1}} \mathcal{W}_{n-1} \xrightarrow{\mathcal{U}_n} \mathcal{W}_n$. Note that, with the except of $\mathcal{W}_0$ and $\mathcal{W}_n$, these intermediate world states are also opaque and unseen from the user's perspective. This system, which we call CoRRPUS, extracts structured information using code-based chain-of-thought–type prompting in order to more accurately track the underlying story state and detect any inconsistencies.

We experiment with three different prompting techniques. The following examples show how the story sentence "Sandra journeyed to the bedroom" would be modified for each prompt type.

- **Comment Only:** This is the simplest prompt, which is given no extra structural information. Thus, it has to rely directly on the comments from the prompt and the story world state initialization. This shows us how well the Code-LLM can infer and fill in information with very little guidance. Example: Given the comment `## Sandra journeyed to the bedroom.`, the model should generate `self.Sandra.location = "bedroom"`.

- **Specific Functions:** This prompt converts each individual sentence into a specific function. In addition to the commented sentences at the beginning of the prompt, the system is also prompted to generate the functions for each sentence. Example: Given the function name `Sandra_journeyed_to_the_bedroom()`, the model should generate a definition for the function, which should contain `self.Sandra.location = "bedroom"`.

- **Abstract Functions:** This last prompting style provides the model with functions for actions or setting attributes. The model then does not have to figure out what it means when a particular event happens but still has to map which function is appropriate for a given story sentence and how to fill in the arguments of the function. Example: Given `go(character, location)` and other functions, generate `go(character=Sandra, destination=bedroom)`.

Full examples of these prompts can be shown in Figure 2 and Appendix 8.1.

4 Experiments

To show how CoRRPUS can improve story understanding via maintaining a world model, we evaluate our system on two tasks: (1) Task 2 of the bAbI set of tasks (Weston et al., 2015)—which tests multi-step reasoning, and (2) Re^3 (Yang et al., 2022), for detecting story inconsistencies in complicated real-world story examples.

4.1 Introduction to the Story Understanding Tasks

bAbI (Weston et al., 2015) is a set of tasks on simple stories that ask questions about what happened during the story. The tasks have various ways of responding to the questions such as with argument relations, supplying supporting acts, or a simple yes/no. Following Nye et al. (2021), we use only bAbI Task 2, which focuses on questions tracking characters carrying objects and moving between different locations. Creswell et al. (2023) also look at Tasks 1, 3, 15, & 16. Since Tasks 1-3 are the same except for differences in story length and Tasks 15 & 16 are much easier tasks, as we found by Creswell et al. (2023)'s results, we decided to focus on Task 2.

Re^3 (Yang et al., 2022) is a story inconsistency detection task aimed at identifying character-based contradictions in stories. Similar to Qin et al. (2019), Yang et al. (2022) built a set of stories, each with a variation that is counterfactual to the original story. These stories were generated by LLMs to generate consistent/contradictory stories based on a story premise and then the model was asked to detect any story inconsistencies. Yang et al. (2022) make use of the Edit function for GPT-3 to correct the detected factual inconsistencies in order to maintain long-range story consistency.

These two tasks have been seen to be extremely challenging for LLMs with naive, few-shot prompting leading to accuracy barely above random chance.

Method	Original Results			Our Results (1-shot)	
	Model	*# Shots*	*Accuracy*	*Model*	*Accuracy*
Random	-	-	-	N/A	25%
GPT-3 (Nye et al., 2021)	GPT-3	0	29.0%	GPT-3	56.5%
Codex w/ natural language	-	-	-	Codex	57.8%
COT (Creswell et al., 2023)	7B LLM	5	∼30%	GPT-3	46.4%
COT (Creswell et al., 2023)	280B LLM	5	∼35%	GPT-3	46.4%
SI (Creswell et al., 2023)	7B LLM	5	∼30%	GPT-3	29.3%
SI (Creswell et al., 2023)	280B LLM	5	Not reported	GPT-3	29.3%
DS (Nye et al., 2021)	GPT-3	10	**100.0%**	-	-
CoRRPUS (comment only)	-	-	-	Codex	67.0%
CoRRPUS (specific functions)	-	-	-	Codex	78.7%
CoRRPUS (abstract functions)	-	-	-	Codex	**99.1%**

Table 1: Accuracy on bAbI task 2. We report other systems' accuracy with the number of examples (# shots) used for their prompts from their respective papers. Chain-of-Thought (COT) and Selection-Inference (SI) prompting are by Creswell et al. (2023) and the Dual-System (DS) is by Nye et al. (2021) using a 5-shot prompt. COT & SI numbers are approximations since they were reported in graph format (Figure 4b of Creswell et al. (2023)). All systems were reimplemented using GPT-3 (175B parameters), with 1-shot prompting to match our experiments.

4.2 Question-and-Answering (bAbI Task 2)

bAbI (Weston et al., 2015) is a question answering dataset for logic-based reasoning tasks. In Task 2, it first provide a story S focusing on the movement of objects and characters throughout the story and a query Q about their locations. The dataset contains 1,000 testing samples of (S, Q, A) tuples, where A is the answer to the query. We choose Task 2 because of the recent neurosymbolic reasoning work evaluated on it (Nye et al., 2021; Creswell et al., 2023), which we use as our baselines.

The CoRRPUS Formulation for bAbI. Starting with the CoRRPUS prompt formulation that was described in Section 3, we first initialize the `character` class with the attributes `name`, `location`, and `inventory`. The object class has attributes `name`, `location`, and `carrier`. Then, we let Codex complete the `World` class by generating the rest of the `story()` function, which tracks the story state changes and ends with a `print()` function that gives the answer to the query Q given in the bAbI Task. We then measure the accuracy of the model selecting the correct answer.

We compare our CoRRPUS system to the following baselines for bAbI Task 2:

1. **Random:** This is just the likelihood of selecting the right multiple choice answer randomly.

2. **GPT-3:** We compare an off-the-shelf GPT-3 davinci model using one-shot prompting to the results from Nye et al. (2021) who use zero-shot prompting.

3. **Codex with Natural Language Prompt:** These are the same natural language prompts as the **GPT-3** baseline but using the Codex model instead. This method will highlight any performance boost from using the Codex model without our CoRRPUS prompts.

4. **LLM with Chain-of-Thought (COT) Prompting (Creswell et al., 2023):** Chain-of-Thought Prompting (Wei et al., 2022) is the process of prompting an LLM to include reasoning traces for solving a given task, and it has been shown to improve model performance on various tasks. Creswell et al. (2023)'s models were a 7 billion- and 208 billion-parameter LLMs from the Gopher family (Rae et al., 2021), and they used 5-shot prompting. In addition to reporting their original accuracy results from their models, we also rerun their experiments using GPT-3 (175 billion parameters) using one-shot prompting to match our experiments.

5. **LLM with Selection-Inference (SI) (Creswell et al., 2023):** This method prompts the LLM to first select sentences relevant to the question, reveal the reasoning chain, and then do the inference. Again, Creswell et al. (2023) use five-shot prompting with the 7 billion- and 280 billion-parameter

models, and we reimplemented the Selection-Inference framework using GPT-3 and one-shot prompting.

6. **GPT-3 Dual-System (DS) (Nye et al., 2021):** This method is based off of System 1/System 2 thinking (Evans, 2003). They specify the functions of all actions used in bAbI Task 2 to create a logical, symbolic component (System 2). Then the system prompts GPT-3 to match each sentence with the actions (System 1) and executes the corresponding function. We did not rerun the results from this experiment.

Results and Discussion. As show in Table 1, even the one-shot prompting CoRRPUS system with Comment Only or Specific Functions achieve 12% and 22% higher accuracy, respectively, compared to vanilla one-shot GPT-3. We believe that is because GPT-3 when used out-of-the-box and unaided is known to be bad at multi-step reasoning (Shridhar et al., 2023; Sap et al., 2022). Meanwhile, Code-LLMs represent knowledge symbolically and therefore are better at logical reasoning (Madaan et al., 2022).

However, the increase in accuracy is not solely due to Codex. When we use the same natural language prompt as the GPT-3 one-shot experiment, the accuracy only goes up 1.3%. It isn't until we introduce the CoRRPUS structured prompting system that we start to see accuracies between 67%-99.1%. We see that when the underlying functions for actions are provided for CoRRPUS (Abstract Functions), it can achieve near-perfect accuracy (99.1%), vastly exceeding other prompting methods. This experiment shows the importance of abstraction to trackability and composability in symbolic reasoning. The other system which approximately matches CoRRPUS in accuracy is the GPT-3 Dual System (Nye et al., 2021), which splits the reasoning steps into two separate systems, uses highly-detailed hand-written rules, and requires 10-shot prompting. We show that providing a simpler one-shot prompt to Codex is enough to perform logic reasoning on these simple stories, as long as it is provided low-level functions to compute over.

With such high accuracies from our CoRRPUS system and Nye et al. (2021)'s Dual System, we are tempted to consider bAbI's Task 2 a solved problem.

4.3 Detecting Story Inconsistencies (Re^3)

Given the simplicity of the sentences in bAbI, we wondered how well CoRRPUS could process and understand stories with complicated real-world sentences, such as those in the Re^3 task. This dataset contains 50 (P, P', S, S') tuples where P denotes a story premise, P' is a premise contradictory to P, S is the story generated from P, and S' is the story generated following P'. The task is framed as classification; one wants to flag (P, S) and (P', S') as consistent and (P, S') and (P', S) as contradictory. They then report the ROC-AUC (area under the ROC curve) score. An example of the comparison between facts of the premise and the story is shown in Figure 3.

CoRRPUS Formulation for Re^3. Following our CoRRPUS prompt starting point (see Section 3), we initialize the `character` class with common person attributes including `name`, `appearance`, `occupation`, `gender`, `age`, and `relations` (to other characters). Then we initialize the main characters in the `World` class and one-shot prompt Codex to complete the `story()` function for tracking the state changes of the characters. Full examples of each of these prompts can be found in Appendix 8.1. Once the `World` state is complete, it is fed into GPT-3 to be converted into natural language text. We then, following the methods of Yang et al. (2022), pass this natural language story state to BART-Large to find any contradictions between the story and the original corresponding premise from the dataset.

We use the following baselines for the Re^3 experiment:

1. **GPT-3:** We query GPT-3 using zero-shot prompting to determine whether there are inconsistencies in the (S, P) pair.

2. **Textual Entailment (Yang et al., 2022):** This method uses the BART-Large-based (Lewis et al., 2020) entailment model trained on MNLI (Williams et al., 2018) to score (S, P) pairs.

3. **Entailment-DPR (Yang et al., 2022):** For each sentence s_i in S, this method computes its relevance score against each sentence in P by using Dense Passage Retrieval (Karpukhin et al., 2020). Then the method takes the sentence with highest relevance score p_i and use the entailment model to detect contradictions.

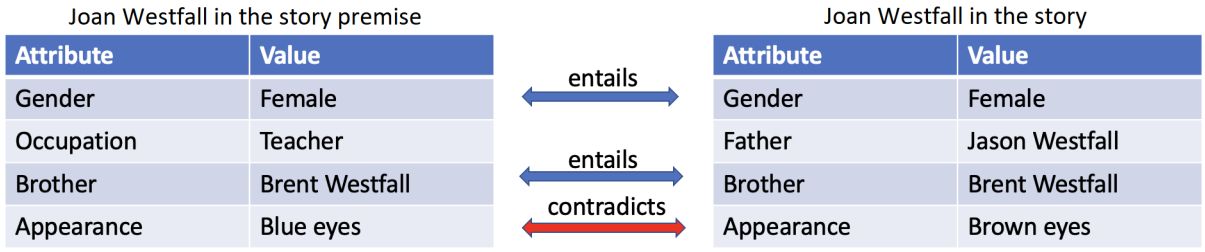

Figure 3: An illustration of the contraction detection process for Re[3]. Following Yang et al. (2022), we check attributes across the premise and the story using a BART-Large-based entailment model and flag any contradictions for attribute-value pairs with the same attribute key. In this example, the "appearance" attribute would be flagged as a contradiction.

4. **Structured-Detect (Yang et al., 2022):** This method prompts GPT-3 to extract an attribute dictionary for each character in the story premise and the story. To prevent hallucinations, the method prompts GPT-3 three times and then uses the BART-Large-based entailment model to take the most-agreed attributes. Finally, the method converts the attribute-value pairs into natural sentences and uses the entailment model to detect contradictions for values under the same key.

Inspired by the Structured-Detect (Yang et al., 2022) and Self-Consistency (Wang et al., 2023) methods, we ask CoRRPUS to complete 3 different generations with temperature equal to 0.7 and select the attribute-value pair by majority voting. Finally, like Yang et al. (2022), we use a BART-Large-based entailment model to flag any contradictions for the values of any attributes found in both the premise and the story. A toy example of this comparison process is shown in Figure 3.

Method	ROC-AUC
Random	0.5
GPT-3	0.52
Yang et al. (2022) baselines:	
Entailment	0.528
Entailment-DPR	0.610
Structured-Detect	0.684
CoRRPUS (comment only)	0.751
CoRRPUS (specific functions)	0.794
CoRRPUS (abstract functions)	0.704

Table 2: ROC-AUC score on the Re[3] consistency detection task. The scores for Entailment, Entailment-DPR, and Structured-Detect models are directly cited from Yang et al. (2022).

Results and Discussion. Table 2 shows that all version of CoRRPUS greatly outperform the baseline methods, with CoRRPUS (specific functions) performing the best at a ROC-AUC score of 0.79.

Subjectively comparing CoRRPUS to GPT-3 shows that GPT-3 tends to struggle with the parsing of the sentences, not knowing what is relevant. Take, for example, the sentence "Mark Woodbury, a middle-aged man with graying hair and a mustache, smiled at Shannon as she walked into his office." CoRRPUS generates

- `self.Mark_Woodbury.appearance.append('graying hair')`
- `self.Mark_Woodbury.appearance.append('mustache')`
- `self.Mark_Woodbury.age.append('middle-aged')`

Meanwhile, GPT-3 generates "Mark is a middle-aged man with graying hair and a mustache." Our particular way of prompting with CoRRPUS uses pre-specified attributes in the `character` class initialization. By pointing out what types of attributes the model should be paying attention to (e.g. appearance or relations), CoRRPUS is better able to extract the relevant information from the natural language sentences of the given story. Meanwhile, GPT-3 ends up summarizing the original sentence.

However, among our three different CoRRPUS prompting methods, we found that their performance are similar, with the Abstract Functions performing the worst and the Specific Functions performing the best. Their similarity in score could stem from the simplicity of the type of information that needs to be reasoned over, namely attributes of characters—with no functions over verbs and

how they unfold. Because of this, the Abstract Functions prompting ends up being a collection of "set" functions (Appendix Figure 6), which is probably making the code more complicated and giving Codex a harder time following it.

However, it's uncertain why Specific Functions perform the best on this task. It could be due to keeping the settings of attributes better separated for each sentence via unique functions (Appendix Figure 5), instead of simply separated by comments (Appendix Figure 4)—since comments are never operated on in real code.

5 Conclusion

We present CoRRPUS, a code-based prompting system that can extract structured information from complicated stories using 1-shot prompting. We conducted experiments on bAbI's Task 2 to show that code-based prompting can leverage symbolic information to perform multi-step logical reasoning better than natural-language–based prompting, regardless of whether a Code-LLM or regular LLM is used. We also evaluate CoRRPUS on detecting story inconsistencies using the Re^3 task (Yang et al., 2022), showing that Code-LLMs can extract relevant information from story sentences better than LLMs. We emphasize that a careful prompting procedure that provides relevant low-level semantic abstractions can greatly improve the accuracy and generalizability of neurosymbolic reasoning but the type of prompt needed is entirely task-dependent.

6 Limitations

We recognize that this work was only performed on two tasks related to story understanding, thus it is difficult to say exactly how robust it really is. However, given the capabilities of LLMs and Code-LLMs, we believe our prompting techniques or similar will prove to be useful to the story understanding community.

Our work also assumes that the CoRRPUS will be asked the same question across stories. In other words, given an example as a prompt, CoRRPUS will follow that example to generate code for the next story. We are not providing the task to CoRRPUS and having it interpret the question to figure out what it should be tracking. We simply tell it to track certain information (e.g., objects, physical features of characters) so that it can solve these tasks. Therefore, for CoRRPUS to work, the user would need to know what information is salient for their task and prompt it to the system.

Even though the Re^3 dataset contains more complicated sentences than bAbI, these are still relatively simple English sentences. We do not know how CoRRPUS would perform on more complex stories or on stories in other languages.

Lastly, there is the issue of access. Due to cost, we were unable to rerun all of the GPT-3 experiments. The pricing of GPT-3 not only hinders new research, but it hinders reproducability efforts such as ours. Furthermore, as of the publication of this paper Codex has been removed from the OpenAI API, and it is as-of-yet unknown if GPT-3.5 or GPT-4 can handle code-based prompting as well. There are, however, still other code-based LLMs available, such as Github's Copilot and Hugging Face's Starcoder.

7 Risks

Working with LLMs is always a risk in itself since they are trained on huge amounts of data, some of which has never been read by developers. This text can include racist, misogynistic, queerphobic, etc. sentiments. Although the risk of harmful text might be reduced in a Code-LLM, comments, variables, and function names might still contain harmful messaging and should always be used with caution, especially when used outside of controlled research settings.

Furthermore, any code that CoRRPUS produces is not guaranteed to run nor is it guaranteed to be completely accurate in its reasoning. However, story understanding is a relatively safe testing space for reasoning and understanding tasks.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant #2030859 to the Computing Research Association for the CIFellows Project.

References

Amal Alabdulkarim, Winston Li, Lara J. Martin, and Mark O. Riedl. 2021. Goal-Directed Story Generation: Augmenting Generative Language Models with Reinforcement Learning. *arXiv preprint arXiv:2112.08593*.

Prithviraj Ammanabrolu, Ethan Tien, Wesley Cheung, Zhaochen Luo, William Ma, Lara J. Martin, and

Mark O. Riedl. 2020. Story Realization: Expanding Plot Events into Sentences. *AAAI Conference on Artificial Intelligence (AAAI)*, 34(5):7375–7382.

Berkeley Andrus, Yeganeh Nasiri, Jay Cui, Ben Cullen, and Nancy Fulda. 2022. Enhanced Story Comprehension for Large Language Models through Dynamic Document-Based Knowledge Graphs. *AAAI Conference on Artificial Intelligence (AAAI)*, 36(10):10436–10444.

Somnath Basu, Roy Chowdhury, Faeze Brahman, and Snigdha Chaturvedi. 2021. Is Everything in Order? A Simple Way to Order Sentences. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 10769–10779. Association for Computational Linguistics.

Faeze Brahman. 2022. *Modeling Key Narrative Elements for Story Understanding and Generation*. Ph.D. thesis, University of California Santa Cruz, Santa Cruz, CA.

Faeze Brahman, Meng Huang, Oyvind Tafjord, Chao Zhao, Mrinmaya Sachan, and Snigdha Chaturvedi. 2021. "let your characters tell their story": A dataset for character-centric narrative understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1734–1752, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Susan Windisch Brown, Julia Bonn, James Gung, Annie Zaenen, James Pustejovsky, and Martha Palmer. 2019. VerbNet Representations: Subevent Semantics for Transfer Verbs. In *First International Workshop on Designing Meaning Representations at ACL 2019*, page 154–163, Florence, Italy. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:1877–1901.

Chris Callison-Burch, Gaurav Singh Tomar, Lara J. Martin, Daphne Ippolito, Suma Bailis, and David Reitter. 2022. Dungeons and Dragons as a Dialogue Challenge for Artificial Intelligence. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 9379–9393. ACL.

Hong Chen, Duc Minh Vo, Hiroya Takamura, Yusuke Miyao, and Hideki Nakayama. 2022. StoryER: Automatic Story Evaluation via Ranking, Rating and Reasoning. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 1739–1753. Association for Computational Linguistics.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating Large Language Models Trained on Code. *arXiv preprint arXiv:2107.03374*.

Matthew Christensen, Jennifer Nelson, and Rogelio Cardona-Rivera. 2020. Using Domain Compilation to Add Belief to Narrative Planners. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 16(1):38–44.

Colin B Clement, Dawn Drain, Jonathan Timcheck, Alexey Svyatkovskiy, and Neel Sundaresan. 2020. PyMT5: multi-mode translation of natural language and Python code with transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 9052–9065, Online.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Antonia Creswell, Murray Shanahan, and Irina Higgins. 2023. Selection-inference: Exploiting large language models for interpretable logical reasoning. *International Conference on Learning Representations (ICLR)*.

Jonathan St. B. T. Evans. 2003. In two minds: dual-process accounts of reasoning. *Trends in cognitive sciences*, 7(10):454–459.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical Neural Story Generation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, page 889–898, Melbourne, Australia. Association for Computational Linguistics.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. 2020. Codebert: A pre-trained model for programming and natural languages. page 1536–1547.

Jacob Garbe, Max Kreminski, Ben Samuel, Noah Wardrip-Fruin, and Michael Mateas. 2019. StoryAssembler: An Engine for Generating Dynamic Choice-Driven Narratives. In *International Conference on the Foundations of Digital Games (FDG)*, page 10.

Artur d'Avila Garcez and Luis C. Lamb. 2020. Neurosymbolic AI: The 3rd Wave. *arXiv preprint arXiv:arXiv:2012.05876*.

Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. 2019. Plan, Write, and Revise: an Interactive System for Open-Domain Story Generation. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL) Demo Track*.

Jian Guan, Zhuoer Feng, Yamei Chen, Ruilin He, Xiaoxi Mao, Changjie Fan, and Minlie Huang. 2022. LOT: A Story-Centric Benchmark for Evaluating Chinese Long Text Understanding and Generation. *Transactions of the Association for Computational Linguistics (TACL)*, 10:434–451.

Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, 8:93–108.

Rujun Han, I-Hung Hsu, Jiao Sun, Julia Baylon, Qiang Ning, Dan Roth, and Nanyun Peng. 2021. ESTER: A Machine Reading Comprehension Dataset for Reasoning about Event Semantic Relations. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 7543–7559. Association for Computational Linguistics.

Rujun Han, Mengyue Liang, Bashar Alhafni, and Nanyun Peng. 2019. Contextualized Word Embeddings Enhanced Event Temporal Relation Extraction for Story Understanding. In *Workshop on Narrative Understanding (NAACL-HLT 2019)*. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations (ICLR)*.

Shanshan Huang, Kenny Q. Zhu, Qianzi Liao, Libin Shen, and Yinggong Zhao. 2020. Enhanced Story Representation by ConceptNet for Predicting Story Endings. In *International Conference on Information and Knowledge Management (CIKM)*, page 3277–3280, Virtual Event, Ireland.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. 2019. CodeSearchNet Challenge: Evaluating the State of Semantic Code Search. *arXiv preprint arXiv:1909.09436*.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. (COMET-)ATOMIC2020: On Symbolic and Neural Commonsense Knowledge Graphs. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, page 6384–6392.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic Detection of Generated Text is Easiest when Humans are Fooled. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, page 1808–1822. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 6769–6781.

Brenden Lake and Marco Baroni. 2018. Generalization without Systematicity: On the Compositional Skills of Sequence-to-Sequence Recurrent Networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2873–2882. PMLR.

Michael Lebowitz. 1984. Creating Characters in a Story-Telling Universe. *Poetics*, 13:171–194.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2022. Deduplicating Training Data Makes Language Models Better. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, volume Volume 1: Long Papers, page 8424–8445, Dublin, Ireland. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, page 7871–7880, Online. Association for Computational Linguistics.

Bryan Li, Lara J. Martin, and Chris Callison-Burch. 2022. CIS²: A Simplified Commonsense Inference Evaluation for Story Prose. In *Workshop on Commonsense Representation and Reasoning (CSRR) at ACL*.

Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. 2022. Language Models of Code are Few-Shot Commonsense Learners. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Lara J. Martin. 2021. *Neurosymbolic Automated Story Generation*. Ph.D. thesis, Georgia Institute of Technology.

Lara J. Martin, Prithviraj Ammanabrolu, Xinyu Wang, William Hancock, Shruti Singh, Brent Harrison, and Mark O. Riedl. 2018. Event Representations for Automated Story Generation with Deep Neural Nets. *Thirty-Second AAAI Conference on Artificial Intelligence*, 32(1):868–875.

Nasrin Mostafazadeh, Aditya Kalyanpur, Lori Moon, David Buchanan, Lauren Berkowitz, Or Biran, and Jennifer Chu-Carroll. 2020. GLUCOSE: GeneraLized and COntextualized Story Explanations. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, volume 4569–4586.

Maxwell Nye, Michael Tessler, Josh Tenenbaum, and Brenden M Lake. 2021. Improving coherence and consistency in neural sequence models with dual-system, neuro-symbolic reasoning. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:25192–25204.

Xiangyu Peng, Siyan Li, Sarah Wiegreffe, and Mark Riedl. 2021. Inferring the Reader: Guiding Automated Story Generation with Commonsense Reasoning. In *Workshop on Narrative Understanding at NAACL-HLT 2021*.

Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. 2019. Counterfactual Story Reasoning and Generation. In *Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, page 5043–5053, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Priyanka Ranade, Sanorita Dey, Anupam Joshi, and Tim Finin. 2022. Computational Understanding of Narratives: A Survey. *IEEE Access*, 10:101575–101594.

Anvesh Rao Vijjini, Faeze Brahman, and Snigdha Chaturvedi. 2022. Towards Inter-character Relationship-driven Story Generation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 8970–8987.

Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. 2020. PlotMachines: Outline-Conditioned Generation with Dynamic Plot State Tracking. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 4274–4295, Online. Association for Computational Linguistics.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. ATOMIC: An Atlas of Machine Commonsense for If-Then Reasoning. *AAAI Conference on Artificial Intelligence (AAAI)*, 33(1):3027–3035.

Maarten Sap, Ronan LeBras, Daniel Fried, and Yejin Choi. 2022. Neural Theory-of-Mind? On the Limits of Social Intelligence in Large LMs. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 3762–3780, Abu Dhabi, United Arab Emirates. ACL.

Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D. Manning. 2019. Do Massively Pretrained Language Models Make Better Storytellers? In *Conference on Computational Natural Language Learning (CoNLL)*, page 843–861, Hong Kong, China.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling Multi-Step Reasoning Capabilities of Large Language Models into Smaller Models via Semantic Decompositions. In *Findings of the Association for Computational Linguistics: ACL 2023*.

Wai Man Si, Prithviraj Ammanabrolu, and Mark Riedl. 2021. Telling stories through multi-user dialogue by modeling character relations. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 269–275, Singapore and Online. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. *AAAI Conference on Artificial Intelligence (AAAI)*, 31(1):4444–4451.

Xiaofei Sun, Zijun Sun, Yuxian Meng, Jiwei Li, and Chun Fan. 2022. Summarize, outline, and elaborate: Long-text generation via hierarchical supervision from extractive summaries. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 6392–6402, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J. Martin, Animesh Mehta, Brent Harrison, and Mark O. Riedl. 2019. Controllable Neural Story Plot Generation via Reward Shaping. In *International Joint Conference on Artificial Intelligence (IJCAI)*, page 5982–5988, Macau, China.

Scott R. Turner and Michael George Dyer. 1985. Thematic knowledge, episodic memory and analogy in MINSTREL, a story invention system. In *Annual Conference of the Cognitive Science Society (CogSci)*, pages 371–375, Irvine, California.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *International Conference on Learning Representations (ICLR)*.

Stephen G Ware and Cory Siler. 2021. Sabre: A Narrative Planner Supporting Intention and Deep Theory of Mind. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, volume 17, pages 99–106.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M. Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. 2015. Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. *arXiv preprint arXiv:1502.05698*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

David Wilmot and Frank Keller. 2021. Memory and Knowledge Augmented Language Models for Inferring Salience in Long-Form Stories. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 851–865, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yuhuai Wu, Albert Qiaochu Jiang, Wenda Li, Markus N Rabe, Charles Staats, Mateja Jamnik, and Christian Szegedy. 2022. Autoformalization with Large Language Models. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Kevin Yang, Yuandong Tian, Nanyun Peng, and Dan Klein. 2022. Re^3: Generating Longer Stories With Recursive Reprompting and Revision. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 4393–4479, Abu Dhabi, United Arab Emirates.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-And-Write: Towards Better Automatic Storytelling. *AAAI Conference on Artificial Intelligence (AAAI)*, 33(1):7378–7385.

Meng-Hsuan Yu, Juntao Li, Zhangming Chan, Dongyan Zhao, and Rui Yan. 2021. Content Learning with Structure-Aware Writing: A Graph-Infused Dual Conditional Variational Autoencoder for Automatic Storytelling. *AAAI Conference on Artificial Intelligence (AAAI)*, 35(7):6021–6029.

Xiyang Zhang, Muhao Chen, and Jonathan May. 2021. Salience-Aware Event Chain Modeling for Narrative Understanding. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 1418–1428, Online and Punta Cana, Dominican Republic.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *International Conference on Learning Representations (ICLR)*.

8 Appendix

8.1 Prompts

These three figures illustrate the comment (Figure 4), specific function (Figure 5), and abstract function (Figure 6) prompts on the Re^3 task for the following story:

The story is set in the present day and takes place in the United States. Joan Westfall is a woman who has died in a car accident. She is a kind and sympathetic person who is eager to help the people she left behind. Brent Westfall is Joan's husband. He is a kind and loving man who is struggling to cope with his wife's death. Jason Westfall is Joan's son. He is a young boy who is struggling to understand his mother's death. Jason Westfall sits on the floor, looking at the empty box that used to hold his sister-in-law's belongings. His gaze is unfocused. his dark blue eyes brimming with tears. He cries for hours, eventually falling asleep from exhaustion. When he wakes up, he feels dazed and ill. Joan died in a car accident on a rainy day several weeks ago. Jason has been carrying on with life ever since as best he can manage, but he still doesn't really know how to cope with Joan's death.

```
## The story is set in the present day and takes place in the United States.
## Joan Westfall is a woman who has died in a car accident. She is a kind and sympathetic person who is eager to help the
    people she left behind.
## Brent Westfall is Joan's husband. He is a kind and loving man who is struggling to cope with his wife's death.
## Jason Westfall is Joan's son. He is a young boy who is struggling to understand his mother's death.
## Jason Westfall sits on the floor, looking at the empty box that used to hold his sister-in-law's belongings.
## His gaze is unfocused. his dark blue eyes brimming with tears.
## He cries for hours, eventually falling asleep from exhaustion.
## When he wakes up, he feels dazed and ill.
## Joan died in a car accident on a rainy day several weeks ago.
## Jason has been carrying on with life ever since as best he can manage, but he still doesn't really know how to cope with
    Joan's death.
## Create a world model state to track each character's appearance, personality, and relations with other characters.

class character:
    def __init__(self, name):
        self.name = name
        self.appearance = []
        self.occupation = []
        self.gender = []
        self.age = []
        self.relations = {}

class World:
    def __init__(self):
        self.Joan_Westfall = character('Joan Westfall')
        self.Jason_Westfall = character('Jason Westfall')
        self.Brent_Westfall = character('Brent Westfall')

    def story(self):
        ## The story is set in the present day and takes place in the United States.
        ## Joan Westfall is a woman who has died in a car accident.
        ## She is a kind and sympathetic person who is eager to help the people she left behind.
        self.Joan_Westfall.gender.append('female')
        ## Brent Westfall is Joan's husband. He is a kind and loving man who is struggling to cope with his wife's death.
        self.Joan_Westfall.relations['husband'] = 'Brent_Westfall'
        self.Brent_Westfall.relations['wife'] = 'Joan_Westfall'
        self.Brent_Westfall.gender.append('male')
        ## Jason Westfall is Joan's son. He is a young boy who is struggling to understand his mother's death.
        self.Joan_Westfall.relations['son'] = 'Jason_Westfall'
        self.Jason_Westfall.relations['mother'] = 'Joan_Westfall'
        self.Jason_Westfall.age.append('young')
        self.Jason_Westfall.gender.append('male')
        ## Jason Westfall sits on the floor, looking at the empty box that used to hold his sister-in-law's belongings.
        self.Jason_Westfall.relations['sister_in_laws'] = 'Joan_Westfall'
        ## His gaze is unfocused. his dark blue eyes brimming with tears.
        self.Jason_Westfall.appearance.append("dark blue eyes")
        ## He cries for hours, eventually falling asleep from exhaustion.
        ## When he wakes up, he feels dazed and ill.
        ## Joan died in a car accident on a rainy day several weeks ago.
        ## Jason has been carrying on with life ever since as best he can manage, but he still doesn't really know how to cope
            with Joan's death.
```

Figure 4: Prompt using CoRRPUS (comment) on Re^3

```
### Create a world model state and track each character's appearance, personality, relationship to other characters, and other
      cruical attributes.
class character:
    def __init__(self, name):
        self.name = name
        self.appearance = []
        self.occupation = []
        self.gender = []
        self.age = []
        self.relations = {}

class World:
    def __init__(self):
        self.Joan_Westfall = character('Joan Westfall')
        self.Jason_Westfall = character('Jason Westfall')
        self.Brent_Westfall = character('Brent Westfall')

    def story(self):
        self.the_story_is_set_in_the_present_day_and_takes_place_in_the_united_states()
        self.joan_westfall_is_a_woman_who_has_died_in_a_car_accident()
        self.she_is_a_kind_and_sympathetic_person_who_is_eager_to_help_the_people_she_left_behind()
        self.brent_westfall_is_joans_husband_he_is_a_kind_and_loving_man_who_is_struggling_to_cope_with_his_wife_s_death()
        self.jason_westfall_is_joans_son_he_is_a_young_boy_who_is_struggling_to_understand_his_mother_s_death()
        self.jason_westfall_sits_on_the_floor_looking_at_the_empty_box_that_used_to_hold_his_sister_in_laws_belongings()
        self.his_gaze_is_unfocused_his_dark_blue_eyes_brimming_with_tears()
        self.he_cries_for_hours_eventually_falling_asleep_from_exhaustion()
        self.when_he_wakes_up_he_feels_dazed_and_ill()
        self.joan_died_in_a_car_accident_on_a_rainy_day_several_weeks_ago()
        self.jason_has_been_carrying_on_with_life_ever_since_as_best_he_can_manage()
        self.but_he_still_doesnt_really_know_how_to_cope_with_joans_death()

    def the_story_is_set_in_the_present_day_and_takes_place_in_the_united_states(self):
        pass

    def joan_westfall_is_a_woman_who_has_died_in_a_car_accident(self):
        pass
    def she_is_a_kind_and_sympathetic_person_who_is_eager_to_help_the_people_she_left_behind(self):
        self.Joan_Westfall.gender.append('female')

    def brent_westfall_is_joan_s_husband_he_is_a_kind_and_loving_man_who_is_struggling_to_cope_with_his_wife_s_death(self):
        self.Joan_Westfall.relations['husband'] = 'Brent_Westfall'
        self.Brent_Westfall.relations['wife'] = 'Joan_Westfall'
        self.Brent_Westfall.gender.append('male')

    def jason_westfall_is_joan_s_son_he_is_a_young_boy_who_is_struggling_to_understand_his_mother_s_death(self):
        self.Joan_Westfall.relations['son'] = 'Jason_Westfall'
        self.Jason_Westfall.relations['mother'] = 'Joan_Westfall'
        self.Jason_Westfall.age.append('young')
        self.Jason_Westfall.gender.append('male')

    def jason_westfall_sits_on_the_floor_looking_at_the_empty_box_that_used_to_hold_his_sister_in_laws_belongings(self):
        self.Jason_Westfall.relations['sister_in_laws'] = 'Joan_Westfall'

    def his_gaze_is_unfocused_his_dark_blue_eyes_brimming_with_tears(self):
        self.Jason_Westfall.appearance.append("dark blue eyes")

    def he_cries_for_hours_eventually_falling_asleep_from_exhaustion(self):
        pass

    def when_he_wakes_up_he_feels_dazed_and_ill(self):
        pass

    def joan_died_in_a_car_accident_on_a_rainy_day_several_weeks_ago(self):
        pass

    def jason_has_been_carrying_on_with_life_ever_since_as_best_he_can_manage(self):
        pass
    def but_he_still_doesnt_really_know_how_to_cope_with_joans_death(self):
        pass
```

Figure 5: Prompt using CoRRPUS (specific function) on Re^3

```
## The story is set in the present day and takes place in the United States.
## Joan Westfall is a woman who has died in a car accident. She is a kind and sympathetic person who is eager to help the
    people she left behind.
## Brent Westfall is Joan's husband. He is a kind and loving man who is struggling to cope with his wife's death.
## Jason Westfall is Joan's son. He is a young boy who is struggling to understand his mother's death.
## Jason Westfall sits on the floor, looking at the empty box that used to hold his sister-in-law's belongings.
## His gaze is unfocused. his dark blue eyes brimming with tears.
## He cries for hours, eventually falling asleep from exhaustion.
## When he wakes up, he feels dazed and ill.
## Joan died in a car accident on a rainy day several weeks ago.
## Jason has been carrying on with life ever since as best he can manage, but he still doesn't really know how to cope with
    Joan's death.
## Create a world model state to track each character's appearance, personality, and relations with other characters.

class character:
    def __init__(self, name):
        self.name = name
        self.appearance = []
        self.occupation = []
        self.gender = []
        self.age = []
        self.relations = {}

class World:
    def __init__(self):
        self.Joan_Westfall = character('Joan Westfall')
        self.Jason_Westfall = character('Jason Westfall')
        self.Brent_Westfall = character('Brent Westfall')

    def set_appearance(self, character, appearance):
        character.appearance.append(appearance)

    def set_occupation(self, character, occupation):
        character.occupation.append(occupation)

    def set_gender(self, character, gender):
        character.gender.append(gender)

    def set_age(self, character, age):
        character.age.append(age)

    def set_relation(self, character, relation, other_character):
        character.relations[relation] = other_character.name

    def story(self):
        ## The story is set in the present day and takes place in the United States.
        ## Joan Westfall is a woman who has died in a car accident. She is a kind and sympathetic person who is eager to help
            the people she left behind.
        self.set_gender(self.Joan_Westfall, "female")
        ## Brent Westfall is Joan's husband. He is a kind and loving man who is struggling to cope with his wife's death.
        self.set_relation(self.Joan_Westfall, 'husband', self.Brent_Westfall)
        self.set_relation(self.Brent_Westfall, 'wife', self.Joan_Westfall)
        self.set_gender(self.Brent_Westfall, "male")
        ## Jason Westfall is Joan's son. He is a young boy who is struggling to understand his mother's death.
        self.set_relation(self.Joan_Westfall, 'son', self.Jason_Westfall)
        self.set_relation(self.Jason_Westfall, 'mother', self.Joan_Westfall)
        self.set_age(self.Jason_Westfall, "young")
        self.set_gender(self.Jason_Westfall, "male")
        ## Jason Westfall sits on the floor, looking at the empty box that used to hold his sister-in-law's belongings.
        self.set_relation(self.Jason_Westfall, 'sister_in_laws', self.Joan_Westfall)
        self.set_relation(self.Joan_Westfall, 'brother_in_laws', self.Jason_Westfall)
        ## His gaze is unfocused. his dark blue eyes brimming with tears.
        self.set_appearance(self.Jason_Westfall, "dark blue eyes")
        ## He cries for hours, eventually falling asleep from exhaustion.
        ## When he wakes up, he feels dazed and ill.
        ## Joan died in a car accident on a rainy day several weeks ago.
        ## Jason has been carrying on with life ever since as best he can manage, but he still doesn't really know how to cope
            with Joan's death.
```

Figure 6: Prompt using CoRRPUS (abstract) on Re^3

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 6

☑ A2. Did you discuss any potential risks of your work?
Section 7

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Sections 1, 3, 4

☑ B1. Did you cite the creators of artifacts you used?
Sections 1, 3, 4

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
These are tools and evaluation metrics free for use within research. We cite them and use them as intended.

☒ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Both GPT-3 and Codex were used to generate text in a responsible manner through controlled experiments. Our use fits within OpenAI's usage policies.

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
The data we use is from pre-existing datasets.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 4

C ☑ Did you run computational experiments?

Section 4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 3

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Sections 3 & 4. Hyperparameter tuning is not applicable since we used off-the-shelf models.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Sections 3 & 4

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 3

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.