Proceedings of the Nineteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2023)

There and Back Again: Extracting Formal Domains for Controllable Neurosymbolic Story Authoring

Jack Kelly, Alex Calderwood, Noah Wardrip-Fruin, Michael Mateas

University of California, Santa Cruz
{jochkell, alexcwd, nwardrip, mmateas}@ucsc.edu

Abstract

Story generators using language models offer the automatic production of highly fluent narrative content, but they are hard to control and understand, seizing creative tasks that many authors wish to perform themselves. On the other hand, planning-based story generators are highly controllable and easily understood but require story domains that must be laboriously crafted; further, they lack the capacity for fluent language generation. In this paper, we explore hybrid approaches that aim to bridge the gap between language models and narrative planners. First, we demonstrate that language models can be used to author narrative planning domains from natural language stories with minimal human intervention. Second, we explore the reverse, demonstrating that we can use logical story domains and plans to produce stories that respect the narrative commitments of the planner. In doing so, we aim to build a foundation for human-centric authoring tools that facilitate novel creative experiences.

Introduction

The recent success of large language models like ChatGPT and GPT-4 has sparked a heated debate about the future of AI in creative work. Amidst the hype and controversy, these systems are already having a material impact on creatives, displacing many contract workers in copywriting, illustration, and graphic design while transforming the workflows of others (Deck 2023). Even where replacing labor with AI is not feasible, the rhetoric of a coming wave of automation is still wielded against creative workers: news sites like Insider and CNET have justified sweeping layoffs through pivots to AI-generated content (Harrison 2023; Sato 2023), while production studios have leveraged the threat of AI against writers in the ongoing WGA strike negotiations (Charity 2023).

As researchers investigating tools to support creative practices, we are troubled by these developments, even as we remain excited by the possibilities these systems have to facilitate new creative experiences. Fundamentally, we are not interested in replacing human writing with AI generated text; we want to encourage individual creativity and provoke authorial reflection and insight. How and to what extent can we use AI for creative support while respecting creative labor and authorial agency? We believe one potential approach is to combine the strengths of language models with those of narrative planners.

Language models have become state-of-the-art in AI for many tasks, including text completion, question-answering, code generation, (short) story generation, and more. These models' general problem solving capacity has given rise to prompt engineering, where domain-specific tasks are performed by reframing them as text completion exercises. Language models perform well with the prompt alone (zero-shot learning) and even better when given a few examples (few-shot learning) (Brown et al. 2020).

But crucially, these models presently lack controllability and explainability. It is hard getting a language model to generate a desired output and much harder to do so consistently. Prompt engineering is often arbitrary and difficult to reason about, with slight variations in wording producing widely divergent outcomes. Further, the voice of any generated text often reverts to a similar style, even when instructed to the contrary (Sawicki et al. 2023). And it is hard to assess why these models make creative decisions, limiting their potential as a tool for exploration while potentially introducing unnoticed bias into the creative process.

An alternative approach to story generation is to use planning techniques; narrative planners use a formal model of a story world to find action sequences that satisfy character and authorial goals. These systems afford substantial creative control, as the author can manipulate the formal model as needed to create a space of potential stories that matches their creative vision. Similarly, it is easy to understand the behavior of these systems; intentional narrative planners like IPOCL (Riedl and Young 2010) can explain the causal reasoning behind their narrative choices.

However, narrative planners aren't designed to generate fluent story text, and they can be difficult to author for. Building and iterating a logical model of a story world is not easy; an author must not only create the entities in the world (characters, objects, settings, etc.) but also define the precise ways these entities can interact, including logical descriptions of state (predicates) and the way the state is changed (operators). While doing so, they must ensure that the system has a sufficient expressive range (Smith and Whitehead 2010) of possible output, i.e., that the system can produce

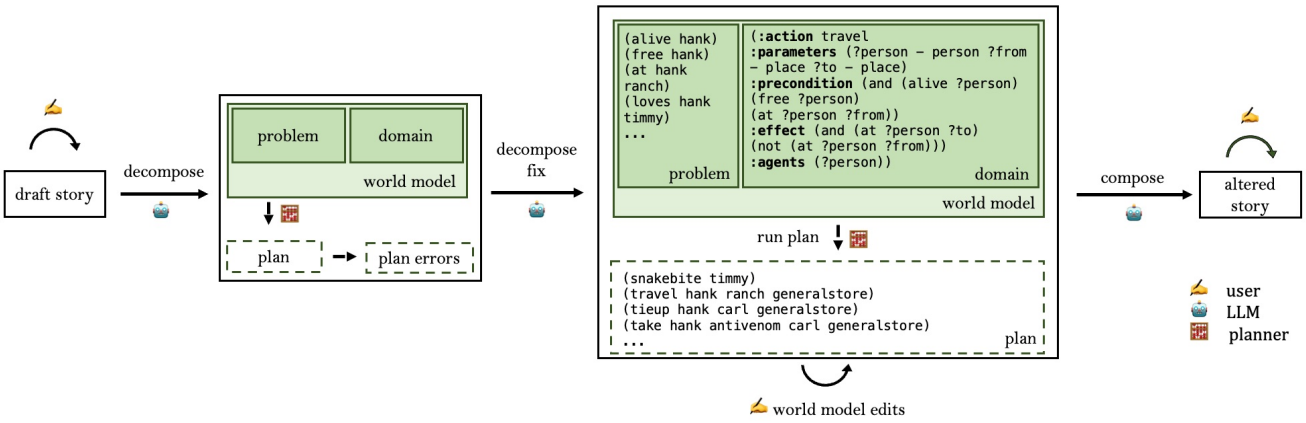

Figure 1: Our human-in-the-loop *Recompose* pipeline assists users with authoring logical story specifications.

interesting and varied stories.

Garbe (2020) describes the authorial burden of narrative systems in terms of a *complexity ceiling* and an *authoring wall*. The complexity ceiling arises when system complexity makes it too difficult to reason about how to add content, while the authoring wall appears when the volume of content the author needs to create is simply too large. Narrative planners can suffer from both problems; it is difficult to reason about how to add new actions and predicates to an already complex domain, and creating content for these systems is quite laborious.

While language models excel at text generation, Kreminski and Martens (2022) note this primarily serves a specific creativity support task, helping authors get unstuck in their writing. They note that different methods may be more suitable for other writing tasks like crafting a story arc. As we work toward making creative support experiences that center authorial agency, we are excited by the expressive affordances of planners and aim to find ways to mitigate their authorial burden. In recent years, researchers have introduced hybrid approaches to story generation that leverage the strength of both neural networks and planning to achieve fluent and coherent stories. We draw on this work to explore a related task: using language models to improve the authorability of narrative planning domains.

In this paper, we introduce a system *Decompose* that extracts a narrative planning domain and problem from a given input story. Next, we outline a system *Compose* which accomplishes the reverse task, taking generated domains and plans to produce representative stories. To assess the performance of *Decompose*, we evaluate it on two story datasets, TinyStories (Eldan and Li 2023) and r/WritingPrompts (Fan, Lewis, and Dauphin 2018), and show that the system can produce domains for a diverse space of input stories. To understand the nature of the generated stories, we conduct thematic analysis, a qualitative analysis method which has been previously adopted to understand story authoring (Halperin and Lukin 2023). We do so first with a set of stories that were generated using a handful of example domains, but then extend this approach to an end-to-end look at the system. As we do so, we discuss the affordances this system has for creative storytelling experiences. Through this work, we hope to provide a foundation for new authoring tools for narrative planners, as well as new creative support systems that center authorial agency.

Related Work

Neurosymbolic Story Generation

In their PhD thesis, Martin (2021) examines the potential of neurosymbolic story generation techniques in detail, exploring its possible use for the RNN-based "automated story writing assistance" proposed by Roemmele (2016). Our system draws on this and other recent work by researchers to guide the output of language models with techniques from planning and symbolic reasoning. A major problem in neural text generation has been maintaining the long term coherence of generated text; while language models continue to improve performance for longer outputs, they have historically struggled with maintaining coherence outside of a local context (Alhussain and Azmi 2022).

One strategy has been to decompose a story into its plot events and surface realization, planning a plot outline and then using it to craft a readable text. Fan, Lewis, and Dauphin (2018) and Yao et al. (2019) used this decomposition with RNNs to achieve improved coherence over previous neural network story systems, and the approach has continued to see success with the transition to transformer models (Mirowski et al. 2023). Subsequent systems have improved coherence by drawing from both a plan and the text generated so far (Ammanabrolu et al. 2019); recently, Tang et al. (2022) extended this further, generating a graph of story possibilities that they traverse conditionally based on the generated text.

Recent approaches have sought to make story generation more controllable and have drawn from planning more explicitly. Peng et al. (2022) used a knowledge graph and a model of reader understanding to guide a language model from an initial prompt to an intended goal state. Ye et al. (2023) introduced a system using a transformer model to implement a variant of Partial Order Causal Link (POCL) planning, building a chain of causal links from an initial

set of conditions to a target story goal. Simon and Muise (2022) conditioned a transformer model directly on a PDDL plan and their system's output to iteratively build a story one action at a time; our system builds on this approach, using GPT-4 with one-shot learning to generate an entire natural language story from a PDDL plan, domain, and problem.

Planning Domain Generation

Our system further builds on a long history of research into deriving planning domains from data. Much of this work has been motivated by situated robotics, with researchers seeking to enable robots to build logical domains they can use to plan their actions. For example, Cresswell, McCluskey, and West (2013) generated logical domains from sequences of robot actions and their observed effects; subsequent work has used a similar framing, seeking to identify a domain from a noisy stream of event logs.

In recent years, researchers have seen increasing success deriving domains from natural language. Sil and Yates (2011) derived preconditions and postconditions for a fixed set of actions given a text input. Yordanova and Kirste (2016) used a stream of valid natural language instructions to learn the corresponding planning domain. Lindsay et al. (2017) used semantic sentence annotations to build action representations from identified subjects, verbs, and objects, creating a generic domain extractor; Hayton et al. (2020) improved upon this approach by additionally reconciling entities across different pronoun references. Recent systems have used language models to achieve higher fidelity output extraction; Jin, Chen, and Zhuo (2022) used a VAE to iteratively improve a domain model extracted from a corpus of thousands of natural language examples. But while extraction systems have continued to improve, their application has been limited by the need for a low complexity of input or a high volume of data.

As large language models like GPT-3 and GPT-4 improve at general-purpose reasoning, researchers have begun to investigate their capacity for symbolic planning. Early results for language model-based planners have been mixed; while researchers have seen limited success with few-shot learning (Huang et al. 2022; Olmo, Sreedharan, and Kambhampati 2021) and fine-tuning (Capitanelli and Mastrogiovanni 2023; Silver et al. 2023), consensus has emerged that language models are not as capable at planning as conventional methods (Valmeekam et al. 2022). But while these models are ill-suited for planning, they have shown promise at domain extraction; Liu et al. (2023a) recently used GPT-4 to extract PDDL domains as part of a natural language planning system. Our system works similarly, using GPT-4 to extract planning domains for the Glaive narrative planner.

System Overview

Our system is divided into two submodules, *Decompose* and *Compose*:

- *Decompose* constructs a logical story representation from a natural language input, then passes the generated model to a state-space planner to construct a sequence of motivated character actions, referred to as the *plan*.
- *Compose* performs the inverse operation, transforming a formal story representation into a natural language telling of the story.

Formal Story Specification

To enforce logical consistency within the stories we generate, we use *Glaive*[1], a Partial Order Causal Link (POCL) planning algorithm which models both *character goals* and *author goals* (Ware and Young 2014). Glaive was designed to balance the competing tensions of *strong story* and *strong autonomy* narrative planners (Riedl and Bulitko 2012). It searches for solutions to a set of predicate-valued author goals, where each step in the plan is an action performed by a character. Further, it ensures that at each step of the generated plan, characters' actions are motivated by an intentional path the system maintains for each character.

Glaive's story planning specifications are described in Planning Domain Definition Language (PDDL), requiring both a *domain* and *problem*. The story specification models the state of the world in terms of logical predicates, and describes the preconditions and effects of actions that may be used to update the world state. Each action is additionally tagged with a set of *agents*: characters who will not select the action unless it is *motivated*, part of an intentional path toward one of the character's goals.

Our system generates story specifications in the PDDL format, allowing it to use Glaive to validity-test the generated planning domains and problems during the *Decompose* step and use them to generate story plans. We use the Glaive-generated plans as an input to our *Compose* step, using them to produce natural language stories.

Language Models and Prompting

Our system uses one-shot prompting of OpenAI's GPT-4 model (gpt-4-0314), following iterative development of the system using GPT-3.5 (gpt-3.5-turbo-0301). GPT-4 makes use of a chat completion API that takes as input a system prompt, followed by a sequence of alternating *assistant* responses and *user* requests. We observe prompt-engineering best practices, informed by recent research literature on prompting methods (Liu et al. 2023b; Kojima et al. 2022; Zhang et al. 2023).

GPT-4 was trained to closely follow instructions given to the model through a *system message*. Our system message contains an explanation of the task, an overview of Glaive's capabilities, and a description of Glaive's dialect of PDDL (the latter two adapted from the Glaive README). We also condition it to avoid problems we observed as we iterated the system (e.g., failing to include closing parentheses). Our system takes advantage of the large token spans GPT-4 can operate on; after substantial revision, our system message grew to an average length of 724.5 words and about 1k model tokens. After the system message, we include a handcrafted input-output pair as a one-shot example. Then we provide the input to be transformed (either a story or a formal story representation, depending on the task). The intent is that the model generates outputs that respect the con-

[1] https://www.cs.uky.edu/~sgware/projects/glaive/

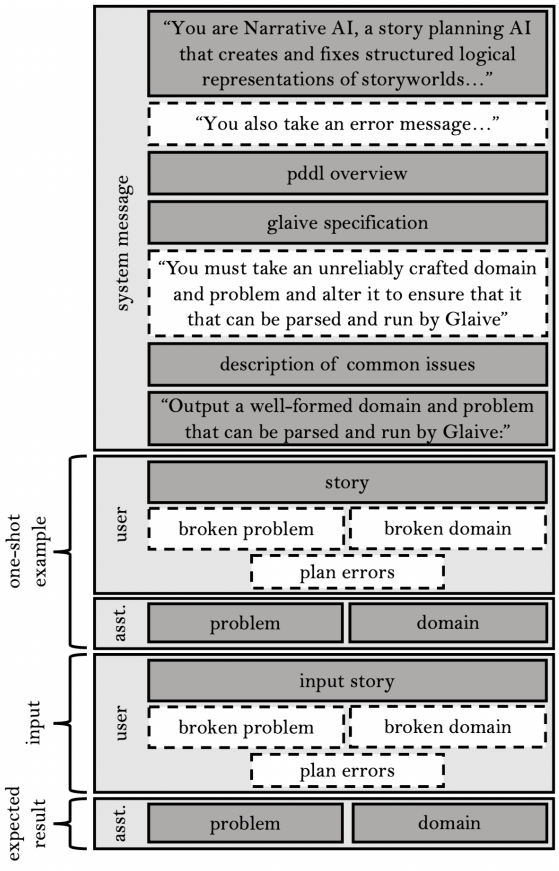

Figure 2: Prompt layout for both phases of our *Decompose* pipeline. Dotted lines represent components that are only present in the second run, after the system has already tried to generate a story world. Each component of the prompt is delimited with a unique start token such as `<|story|>`.

straints outlined in the system message while also mirroring the form of the one-shot example. The full system messages and one-shot examples are available on Github[2].

Decompose (*There*)

Decompose transforms a natural language story into our formal story specification, using the prompt template described in Figure 2.

First, it uses a one-shot prompt to generate a domain and problem from a given story. The system passes the output to Glaive for plan generation. If the plan generation fails for any reason, we try again with a second one-shot prompt, this time conditioning on both the failed domain and problem and the full error message from Glaive; for example:

```
"(alive ?eaten)" could not be parsed as
Expression (at line 60))
```

If Glaive can compile the problem and initiate planning, but no solution is found before a timeout, we provide a hand-crafted message specifying issues that were common at this stage and how to resolve them.

This technique, known as automated debugging (Silver et al. 2023), showed early promise in our iteration cycle and was central in getting the early versions of the system to generate plans.

In our experiments, we only perform a single re-run through the pipeline to save on costs and time. However, the system is designed to allow collaborative human-in-the-loop editing and multiple passes through the auto-debugging loop.

Each prompt in the *Decompose* step uses a model temperature of 0.2 to encourage consistent and well-formed PDDL output.

Compose (*Back*)

The *Compose* pipeline is structured similarly to *Decompose*, with a prompt that mirrors the instructions in Figure 2. The system message states the task is to produce "flash fiction narratives". It ends with the instruction:

```
You will produce a one-page story that is
logically consistent with a hypothetical
story plan produced by the domain and
problem. It should use the provided domain
as an ontology from which to base the events
 in the story. The initial conditions should
 be consistent with those in the problem.
```

As in Figure 2, we use a one-shot pipeline to generate stories from our formal representations. We created two versions of the *Compose* pipeline. The first uses an example problem and domain without a Glaive-generated plan (*without_plan*); the second includes a corresponding plan that Glaive generates from the pair (*with_plan*).

We expect each pipeline to have different controllability characteristics, and we built both to explore their affordances within a co-creative system. Both are structured similarly, using PDDL as a formal representation of a story world; the predicates describe story entities and their properties, while the actions denote the events that should be central to the plot. The *with_plan* pipeline is given a straightforward story outline, so we expect it to weave narrative ties between a closely ordered chain of events. The *without_plan* pipeline must find a narrative solution to the author goals without a clearly defined narrative structure, so we expect it to generate stories with greater narrative variety at the expense of controllability. As both versions use a one-shot pipeline, we expect the generated stories to be guided by the structure and style of the included example.

We used a temperature setting of 0.9 for all experiments to encourage creativity.

Recompose (*There and Back Again*)

While our system is capable as both a story world modeler and a story generator, it is motivated by a desire to combine these two modules into a single system (Figure 1); by doing so, we intend to build the foundation for an authoring tool for logical story specifications as well an intelligent user interface for nonlinear story writing.

[2]https://github.com/alex-calderwood/there-and-back

For domain authoring, we envision that users will edit story specifications through a combination of text entry and drag-and-drop textlet composition. Generated stories may be used as an indication of the expressive range of the draft story world; these stories could also act as inspiration for the modification or addition of domain operators.

We also believe this configuration is well suited to building a *narrative instrument* in the mold of Kreminski et al. (2022), which utilizes narrative awareness in a dynamic interactive storytelling interface. Kreminski et al. uses story sifting to help a writer/player discover plot directions that best resolve chosen story goals. We propose a system that generates trees of branching stories, each representing alternate possible worlds. Rather than use story generation to replace human writing, we imagine a tool where these stories are used to provoke defamiliarization and plot-structure ideation (Kreminski and Martens 2022; Gero et al. 2022). A language model alone is not suited for this task, as it demands substantial authorial control; a conventional planner without our proposed enhancements makes the authorial burden too large.

Our prototype, *Recompose*, begins with the user writing a roughly half-page story they can later revise. From this, a story specification is generated with *Decompose*. At this stage, the generated logical representation is not final but a working draft of a story world subject to authorial revision. From here, an author can correct errors and revise the story specification. Each iteration of the story is accompanied by a call to Glaive to generate new potential story plans, and each plan is then used to generate stories using *Compose*.

In future work, we intend to build a more sophisticated interface for this system and validate it with a user study; for now, we use it to motivate our work and to ground our evaluation.

Evaluation

We evaluate our system through a mixture of quantitative and qualitative analysis. First, we use quantitative methods to show that our domain generation can create domains from a diverse set of plausible user inputs. Then, we perform qualitative thematic analysis to understand the qualities of our story output and how it may function in a co-creative system.

Quantitative Analysis

To test the *Decompose* step, we evaluate it on two story datasets, TinyStories (Eldan and Li 2023) and r/WritingPrompts (Fan, Lewis, and Dauphin 2018). The TinyStories dataset consists of short GPT-4 generated stories intended to represent a lower-grade writing level; while the stories are simple and potentially favorable to logical formalization, they are also odd and often incoherent, creating a challenge for the system to construct a meaningful story domain. In contrast, the r/WritingPrompts dataset consists of stories scraped from Reddit; these stories vary considerably in length and writing quality but generally have complex plots drawn from sci-fi and horror. Taken together, we believe these offer a considerable modelling challenge while reflecting the breadth of inputs we intend to accommodate with our tool.

Each story was run through the *Decompose* pipeline. As a simple ablation study, we test the system with GPT-3.5 in addition to GPT-4, and test the system with and without the auto-debugging step. For each generated domain and problem, we check if it can be compiled by Glaive. If so, we additionally test if they can be used to generate a plan successfully. (While a domain and problem may be well-formed, the need for plans to satisfy both character intentions and author goals means many problems cannot be solved.) We report the results for this task in Table 2.

But while a generated story specification may compile and generate a plan, that alone does not tell us the quality of the results. Consequently, we use a custom PDDL parser to analyze the structure of our generated outputs; we report several representative statistics about the story specifications to assess the output's complexity, including the average number of actions, the number of predicates in the goal, and the number of steps in the generated plan. These results are reported in Table 1. Additionally, we include statistics for a baseline set of human-authored domains selected from the Glaive release (*Ark*, *Fantasy*, *Space*, and *Western*).

Results

While it is hard to assess without a concrete baseline, we believe our results show that our system offers state-of-the-art versatility and capability in domain and problem generation.

Our system produces story specifications that can create plans. First, in Table 2, we can see that the full *Decompose* pipeline produces results that compile 77% of the time and result in plans 34% of the time. Given that we intend to use this in a human-in-the-loop tool, this is performative enough to enable users to easily craft story planning specifications. In an interactive process, errors can be corrected; most of the time, the user will not need to change anything. Interestingly, the performance is similar across the two datasets, suggesting the limit is more GPT-4's ability to reason about planning logic than it is to abstractly reason about the stories. As expected, the performance declines across our ablation; GPT-4 still performs well without auto-debugging but is less consistent, and the system cannot produce anything with only GPT-3.5.

The story specifications are sophisticated. Second, while the generated story specifications are notably less complex than the handcrafted examples we use as a baseline, we can see in Table 1 that they are still sophisticated. While it varies across the different data slices, we can see that generated domains generally consist of 4-6 predicates and actions, each with about 1-2 parameters. While the expressive range of these specifications is likely narrow, these results suggest a solid foundation a user can iterate on. Notably, the specifications that generate plans are consistently less complicated than the ones that compile but do not plan. This gap, while small, likely reflects how increasing the constraints on a story space can make problems harder to solve. Further, it suggests GPT-4 may be less capable when dealing with more ambitious domain constructions.

dataset	plan?	total domains	avg. constants	avg. actions	avg. predicates	avg. types	avg. axioms	avg. precond./action	avg. effect/action	avg. agents/action	avg. params./action	avg. params./predicate	avg. objects	avg. inits	avg. intents	avg. goals	avg. steps
tiny stories	plan	33	0.09	4.42	4.97	1.82	0.03	1.64	1.40	1.02	1.75	1.32	3.73	6.61	3.82	3.00	3.48
	no plan	43	0.05	4.93	5.58	1.79	0.07	1.67	1.29	1.08	1.76	1.43	4.12	7.33	4.05	3.70	
writing prompts	plan	34	0.06	4.41	6.12	1.76	0.15	1.99	1.18	0.91	1.55	1.27	4.12	7.41	3.65	2.97	4.32
	no plan	44	0.02	5.23	6.89	2.20	0.16	2.29	1.26	0.98	1.86	1.36	6.18	10.93	4.07	4.18	
combined	plan	67	0.07	4.42	5.55	1.79	0.09	1.81	1.29	0.97	1.65	1.29	3.93	7.01	3.73	2.99	3.91
	no plan	87	0.03	5.08	6.24	2.00	0.11	1.99	1.28	1.03	1.81	1.39	5.16	9.15	4.06	3.94	
glaive examples		4	0.75	8.0	9.75	4.0	0.5	4.59	4.25	0.94	2.72	1.59	6.5	20.0	5.75	3.25	8.0

Table 1: Average element counts for generated domains, problems and plans. The complexity is similar across the two datasets; domains that generate plans tend to be lower complexity. The human-authored Glaive examples are consistently more complex.

model	dataset	compile rate	plan rate
gpt-3.5-turbo	combined	0.00	0.00
gpt-4	tiny stories	0.62	0.24
	writing prompts	0.53	0.18
	combined	0.57	0.21
gpt-4, autodebug	tiny stories	0.76	0.33
	writing prompts	0.78	0.34
	combined	0.77	0.34

Table 2: While the system fails to generate domains that compile using GPT-3.5, it succeeds at generating story planning specifications that both compile and plan with GPT-4. The auto-debugging version of our system succeeds more often at both tasks.

Thematic Analysis

To help understand the characteristics of our generated stories, we conduct thematic analysis paired with close reading. Thematic analysis is a systematic method in which data is read and annotated (*familiarization*) in order to develop descriptive *codes* that stand for observed features in the data. These codes are used to iteratively construct overarching *themes* that capture insight related to one's research question (Braun and Clarke 2012).

While we recognize that quantitative stylistics (Roemmele, Gordon, and Swanson 2017; Purdy et al. 2018) have proven invaluable in understanding nuanced features of narrative writing, we are acutely aware that such methods are fundamentally limited in their ability to evaluate stories due to the necessity of human interpretation within the sense-making process (Ramsay 2011). Thematic analysis is an alternative, reflexive approach that has been recognized as a valid method for interpreting stories (Halperin and Lukin 2023); we choose it here as we believe this gives us a nuanced view into the practical uses of our generated stories in a creative tool. We performed two sets of analyses: first on stories generated using *Compose* with handcrafted domains, and then on the end-to-end output of automated runs of *Recompose*.

Compose - Initial Themes

We conducted the first round of analysis on *Compose* output using the *Ark*, *Fantasy*, *Space*, and *Western* domains. For each domain, we wrote a short (roughly 400-word) story that narrativizes the events of the corresponding plan using the actions and predicates of the formal model. We used each pairing of these stories as an example and an input, leading to 4×3 unique permutations, each of which had a *with_plan* and *without_plan* version. Using this small corpus, we performed qualitative coding and theme identification, seeking to understand the stories holistically. The following are the themes that emerged through this process:

Generated stories closely followed the plan but contained narrative gaps. The *with_plan* pipeline generated stories that closely followed the generated plan and contained all the necessary character actions but still missed typical elements common in storytelling. In one story, Vince, a rich suitor, failed to win the heart of Talia. While the victorious suitor had a happy ending, the narrative left Vince 'dangling' without any further interactions with other characters. In another case, Will, the sheriff of an unusually rattlesnake-infested town, was introduced to no effect. In many stories, key events are glossed over with a sentence ("Indiana managed to defeat the Nazis").

This suggests that while our system is successfully generating narratives according to instruction, it may be beneficial to integrate rhetorical planning (Mann and Thompson 1987) or formal models of reader experience management, such as narrative tension arcs (Chung et al. 2022).

Excluding plans led to unfulfilled author goals. Our *without_plan* pipeline succeeded in producing tales whose central events matched actions in the story domain but sometimes failed to arrive at a story that met the authorial goals provided in the problem. In one instance, author goals specified that the character Zoe and the lizard guardian of a planetary surface were meant to engage in a reluctant battle:

```
(and (not (habitable surface))
     (friends zoe lizard)
     (not (alive lizard))
```

The *without_plan* story instead concludes, "The cosmic

duo was unstoppable as they explored the unknown reaches of the universe." The *with_plan* version ends, "Zoe mourned the loss of her friend Lizard, who tragically perished during the eruption," after she left it stunned near a volcano.

Stories reproduced the style of the example narratives. We wrote our example narratives to match their corresponding toy Glaive domain; for example, we matched abstract plan steps like:

```
(take nazis ark indiana usa)
(open-ark nazis)
```

with:

```
They took the ark from him at gunpoint, and
took it to a secret location. They opened it
, and were immediately killed by the wrath
of God.
```

While we intended to experiment with more stylized texts, each of our example stories used a similar declarative voice.

One point of interest that emerged in our close reading was the degree to which these examples guided the style of the generated stories. Stories generated in our experiment sample were uniform along at least four of the most salient dimensions of narration (including voice, focalization, reliability, and narrative distance) (Abbott 2020). They were told in reliable, third person, and non-character styles, in goal-oriented declarative sentences.

Text was fluent and often engaging. The text, though straightforward and lacking personal style, was clear and engaging, with phrases deploying simple modifiers effectively: "unearthing the Ark from its sandy tomb" or "the venom in Timmy's veins took its toll, and the young boy passed away before his father could administer the cure."

Text was often unexciting. On the flip side, the generated texts were often uninspired; two persistent weaknesses identified in our coding were the frequency with which they overused modifiers and a tendency toward a cloyingly melodramatic voice. For example, stories included descriptions of "the notorious Nazis" and mentioned that characters "desperately wanted Talia to be both happy and prosperous." We believe this reflects the stylistic simplicity of our canonical stories and their rigid plot adherence.

Stories successfully threaded between plan steps. Generated plans sometimes put characters into situations without straightforward narrativizations. But despite these confusing sequences, the *with_plan* system was able to infill transitions between disparate actions. In one instance, the sheriff's later absence from a plan is motivated, despite the character's intention to enact justice on lawbreakers:

```
Once Sheriff Will heard about Timmy's
unfortunate death and Hank's own snakebite,
he decided not to pursue the case. He knew
that Hank's actions had been driven by
desperation and a father's love.
```

Stories made direct reference to the formal model. As in the previous example, the stories often included motivations directly lifted from the input plan. We found it commonly asserted in the stories that actions were taken according to a character's "intention" or that an action they took satisfied their goal, seemingly deriving this language from the variable names in the planning specification.

Evaluating the Entire System

To evaluate the full system on texts written in styles that may better align with our expressive goals, we used *Recompose* to generate formal representations and narratives for 100 samples from r/WritingPrompts and TinyStories. For each, we used the *Ark* story as the one-shot example and used each version of *Compose*. We randomly selected 12 successful examples for close reading; for each one, we analyzed the formal story specification and both generated stories.

Compose - Additional Themes

Our initial themes carried into the stories generated from the end-to-end system, but new themes also emerged:

Story objects and characters were referenced directly. In general, the system did not introduce hypernyms for referenced objects but rather used variable names from the domain, such as "Poem1". Characters also were often referenced directly as they appeared in the plan or problem: "Reader1" and "Friend1".

Strong preference for an external voice. The r/WritingPrompts dataset is characterized by personal writing with internal voice, clashing with the voice of our one-shot example. First-person pronouns were consistently replaced with characters like "Narrator" or "Protagonist".

Divergent plans still lead to coherent stories. Glaive occasionally solved *Decompose*-generated problems with character action sequences that did not match those of the original story. When this occurred, the *with_plan* pipeline typically followed the generated plan. But these domains were often particularly expressive for *without_plan* as well, with the stories integrating logically sensible domain actions that the plans did not use. Both systems produced coherent stories, but *without_plan* introduced creative possibilities using the LLM, while *with_plan* did so with the planner.

Decompose - Themes

Actions and predicates correspond to the story. In most of the system-generated story models, the actions, predicates, and other objects cohered with the story. For example, one short story involved a group of children enjoying themselves while organizing a room, reading books, and smelling soap. The system generates predicates appropriately:

```
(:predicates
  (at ?character - character ?place - place)
  (has ?character - character ?item - item)
  (on-shelf ?item - item)
  (organized ?place - place)
  (reads ?character - character)
  (happy ?character - character))
```

Actions in this model allow characters to find objects and put objects they have on a shelf. Reading books and smelling soap both make them happy, e.g.:

```
;; A character reads a book.
(:action read
  :parameters
    (?character - character)
  :precondition
    (reads ?character)
  :effect
    (happy ?character)
  :agents
    (?character))
```

The generated solution does not make use of the smell-soap action likely because *read* duplicates the effect of making the agent happy:

```
(:goal
    (and (organized house)
    (happy lily)
    (happy friend1)
    (happy friend2)))
(define (plan organize-room)
  (:problem organize-room)
  (:steps (read friend2)
          (read friend1)
          (read lily)
          (find lily book1)
          (put-on-shelf lily book1 house)))
```

Story models appropriately match abstraction level to the story. The system can generate story representations with varied concerns: domains may describe internal emotions and abstract character actions. An example from the r/WritingPrompts dataset has little by way of plot, rather is a meditation on an aging face:

```
(:steps
    ...
    (notice-change protagonist)
    (experience-emotion protagonist wisdom)
    (memorize-appearance protagonist))
```

In this example, actions represent emotional changes in an interior mental state rather than moving objects or acting on a space of multiple characters. In another evocative example, a writer gives a poem life; the actions in the generated domain define abstract moments, such as the poem dying after losing its power.

Problem overspecifies the solution. In multiple examples, the domain creates a predicate to mark that an event has occurred. The problem then uses these event predicates in the author goals to directly assert every action that should take place. While this does lead to successful solutions, it indicates that the domain has been narrowly constructed for a single story rather than a space of stories. This lack of expressivity could be problematic in a creative tool and suggests an area for future improvement.

Limitations & Future Work

Fundamentally, we do not want to abdicate the responsibility for the potential of AI systems to do harm. At each stage of designing this system, we have sought to emphasize authorial control and creative opportunity, hoping to present a positive vision for these tools; it remains to be seen how reasonable that vision is.

This work presents a variety of opportunities for further extension, and we intend to continue to pursue the potential suggested by this prototype.

Though we stand by close reading as a method for elucidating an understanding of the stories we generate, we intend to perform a reader study paired with computational textual analysis in follow up work to make our insights here more rigorous.

Further, while our pipeline is successful in the aggregate, aspects of it are somewhat arbitrary, achieved through error-prone iteration. In future work, we intend to more thoroughly explore different design choices and how they impact the generated output, including the choices of model, prompts, one-shot examples, and planner. For example, using the newer Sabre planner (Ware and Siler 2021) may improve results when it becomes publicly available.

As we elaborated in our *Recompose* section, we are excited about the possibility of these techniques to enable new co-creative systems. We are currently developing an authoring tool for PDDL narrative domains that uses our system; while our system has been powerful in offline use, we want to explore its potential within a reactive user interface. Similarly, we plan on integrating this system into an expressive storytelling interface that has characteristics of both a writing tool and a storytelling game.

Conclusion

In this paper, we demonstrated the potential for language models to work alongside planners to craft both narratives and logical story models.

With *Decompose*, our system can be used to produce editable drafts of narrative planning specifications based on input stories. These input stories need not be limited to a single narrative style or thematic scope. Because of its generality, our system design shows promise as an intelligent backbone for future plan-based authoring in which authors retain creative control. Still, work must be done to improve quality and consistency.

For *Compose*, generated stories were fluent and narratively coherent, but stylistically straightforward and highly influenced by the one-shot training example. They follow the plans generated by the narrative planner and adhere to the logical story domains. Their rhetorical simplicity indicates that other plan-based storytelling methods, like rhetorical planners, might have use in LLM story generation.

Together, these systems suggest a future for creative support tools that combine language models and narrative planning to facilitate human-centered authorship. This work may lead to interfaces that ease the authorial burden of narrative PDDL domains and enable new creative instruments to support storytellers.

Appendix - Example Output

The following is a *Decompose* output from the TinyStories dataset selected to be representative of the trials that exhibited successful planning; more are available on our Github.

Input Story

Once upon a time, there was a little boy named Tom. He lived in a deep forest with his mom and dad. Tom liked to surprise his friends with gifts. One day, Tom found a big, pretty stone. He thought it would make a great surprise gift for his best friend, Sue. Tom hid the stone behind his back and went to find Sue. When he saw her, he said, "Sue, I have a surprise for you!" Sue's face had a big smile. Tom showed her the stone, and Sue loved it. They played with the stone all day, rolling it and making funny shapes in the dirt. As the sun went down, Tom and Sue were tired but happy. They learned that giving surprises can make others smile and feel good. The deep forest was filled with the joy of their laughter. And from that day on, Tom and Sue always tried to surprise each other with little gifts, spreading smiles and happiness wherever they went.

Generated Story Representation

```
(define (domain surprise-gifts)
  (:requirements :adl :domain-axioms
     :intentionality)
  (:types character - object place - object
     item - object)
  (:predicates
    (at ?character - character ?place -
       place)
    (has ?character - character ?item - item
       )
    (happy ?character - character)
    (gave-gift ?giver - character ?receiver
       - character ?item - item))

  ;; A character finds an item.
  (:action find
    :parameters
      (?character - character ?item - item ?
         place - place)
    :precondition
      (and (at ?character ?place)
      (not (has ?character ?item)))
    :effect
      (has ?character ?item)
    :agents
      (?character))

  ;; A character plays with an item.
  (:action play
    :parameters
      (?character - character ?item - item ?
         place - place)
    :precondition
      (and (at ?character ?place)
      (has ?character ?item))
    :effect
      (happy ?character)
    :agents
      (?character))

  ;; A character gives an item to another
     character.
  (:action give
    :parameters
      (?giver - character ?item - item ?
         receiver - character ?place -
         place)
    :precondition
      (and (not (= ?giver ?receiver))
      (at ?giver ?place)
      (has ?giver ?item)
      (at ?receiver ?place))
    :effect
      (and (not (has ?giver ?item))
      (has ?receiver ?item)
      (gave-gift ?giver ?receiver ?item))
    :agents
      (?giver ?receiver))

  ;; A character makes another character
     happy.
  (:action make-happy
    :parameters
      (?character - character ?target -
         character)
    :precondition
      (and (not (= ?character ?target))
      (not (happy ?target)))
    :effect
      (happy ?target)
    :agents
      (?character)))

(define (problem surprise-gifts)
  (:domain
    surprise-gifts)
  (:objects
    tom sue - character
    forest - place
    stone - item)
  (:init
    (at tom forest)
    (at sue forest)
    (intends tom (gave-gift tom sue stone))
    (intends tom (happy tom))
    (intends tom (happy sue))
    (intends sue (happy sue))
  )
  (:goal
    (and (gave-gift tom sue stone)
    (happy tom)
    (happy sue))))
```

Glaive Plan

```
(define (plan surprise-gifts-solution)
  (:problem surprise-gifts)
  (:steps (find sue stone forest)
          (find tom stone forest)
          (play tom stone forest)
          (give tom stone sue forest)
          (play sue stone forest)))
```

References

Abbott, H. P. 2020. *The Cambridge introduction to narrative*. Cambridge University Press.

Alhussain, A. I.; and Azmi, A. M. 2022. Automatic Story Generation: A Survey of Approaches. *ACM Computing Surveys*, 54(5): 1–38.

Ammanabrolu, P.; Tien, E.; Cheung, W.; Luo, Z.; Ma, W.; Martin, L.; and Riedl, M. 2019. Guided Neural Language Generation for Automated Storytelling. In *Proceedings of the Second Workshop on Storytelling*, 46–55. Florence, Italy: Association for Computational Linguistics.

Braun, V.; and Clarke, V. 2012. *Thematic analysis.* American Psychological Association.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 1877–1901. Curran Associates, Inc.

Capitanelli, A.; and Mastrogiovanni, F. 2023. A Framework to Generate Neurosymbolic PDDL-compliant Planners. Accessed: 2023-08-15, arxiv:2303.00438.

Charity, J. 2023. Hollywood's Double-Edged Reckoning With AI. https://www.theringer.com/tv/2023/7/21/23802591/writers-strike-2023-wga-sag-aftra-ai-artificial-intelligence-actors. Accessed: 2023-08-15.

Chung, J. J. Y.; Kim, W.; Yoo, K. M.; Lee, H.; Adar, E.; and Chang, M. 2022. TaleBrush: visual sketching of story generation with pretrained language models. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*, 1–4.

Cresswell, S. N.; McCluskey, T. L.; and West, M. M. 2013. Acquiring Planning Domain Models Using LOCM. *The Knowledge Engineering Review*, 28(2): 195–213.

Deck, A. 2023. The workers at the frontlines of the AI revolution. https://restofworld.org/2023/ai-revolution-outsourced-workers/. Accessed: 2023-08-15.

Eldan, R.; and Li, Y. 2023. TinyStories: How Small Can Language Models Be and Still Speak Coherent English? arXiv:2305.07759.

Fan, A.; Lewis, M.; and Dauphin, Y. 2018. Hierarchical Neural Story Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 889–898. Melbourne, Australia: Association for Computational Linguistics.

Garbe, J. 2020. *Increasing Authorial Leverage in Generative Narrative Systems*. Ph.D. thesis, UC Santa Cruz.

Gero, K.; Calderwood, A.; Li, C.; and Chilton, L. 2022. A Design Space for Writing Support Tools Using a Cognitive Process Model of Writing. In *Proceedings of the First Workshop on Intelligent and Interactive Writing Assistants (In2Writing 2022)*, 11–24. Dublin, Ireland: Association for Computational Linguistics.

Halperin, B. A.; and Lukin, S. M. 2023. Envisioning Narrative Intelligence: A Creative Visual Storytelling Anthology. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23. New York, NY, USA: Association for Computing Machinery. ISBN 9781450394215.

Harrison, M. 2023. Insider Lays Off 10 Percent of Staff After Announcing Pivot to AI. https://futurism.com/insider-layoffs-ai. Accessed: 2023-08-15.

Hayton, T.; Porteous, J.; Ferreira, J.; and Lindsay, A. 2020. Narrative Planning Model Acquisition from Text Summaries and Descriptions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(02): 1709–1716.

Huang, W.; Abbeel, P.; Pathak, D.; and Mordatch, I. 2022. Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents. In Chaudhuri, K.; Jegelka, S.; Song, L.; Szepesvari, C.; Niu, G.; and Sabato, S., eds., *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 9118–9147. PMLR.

Jin, K.; Chen, H.; and Zhuo, H. H. 2022. Text-Based Action-Model Acquisition for Planning. arxiv:2202.08373.

Kojima, T.; Gu, S. S.; Reid, M.; Matsuo, Y.; and Iwasawa, Y. 2022. Large Language Models are Zero-Shot Reasoners. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 22199–22213. Curran Associates, Inc.

Kreminski, M.; Dickinson, M.; Wardrip-Fruin, N.; and Mateas, M. 2022. Loose Ends: A Mixed-Initiative Creative Interface for Playful Storytelling. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 18(1): 120–128.

Kreminski, M.; and Martens, C. 2022. Unmet Creativity Support Needs in Computationally Supported Creative Writing. In *Proceedings of the First Workshop on Intelligent and Interactive Writing Assistants (In2Writing 2022)*, 74–82. Dublin, Ireland: Association for Computational Linguistics.

Lindsay, A.; Read, J.; Ferreira, J.; Hayton, T.; Porteous, J.; and Gregory, P. 2017. Framer: Planning Models from Natural Language Action Descriptions. *Proceedings of the International Conference on Automated Planning and Scheduling*, 27: 434–442.

Liu, B.; Jiang, Y.; Zhang, X.; Liu, Q.; Zhang, S.; Biswas, J.; and Stone, P. 2023a. LLM+P: Empowering Large Language Models with Optimal Planning Proficiency. arxiv:2304.11477.

Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; and Neubig, G. 2023b. Pre-Train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Comput. Surv.*, 55(9).

Mann, W. C.; and Thompson, S. A. 1987. *Rhetorical Structure Theory: Description and Construction of Text Structures*, 85–95. Dordrecht: Springer Netherlands. ISBN 978-94-009-3645-4.

Martin, L. J. 2021. *Neurosymbolic Automated Story Generation.* Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA, USA.

Mirowski, P.; Mathewson, K. W.; Pittman, J.; and Evans, R. 2023. Co-Writing Screenplays and Theatre Scripts with Language Models: Evaluation by Industry Professionals. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 1–34. Hamburg Germany: ACM. ISBN 978-1-4503-9421-5.

Olmo, A.; Sreedharan, S.; and Kambhampati, S. 2021. GPT3-to-plan: Extracting Plans from Text Using GPT-3. In *Proceedings of the 2021 ICAPS Workshop on Knowledge Engineering for Planning and Scheduling*.

Peng, X.; Xie, K.; Alabdulkarim, A.; Kayam, H.; Dani, S.; and Riedl, M. 2022. Guiding Neural Story Generation with Reader Models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, 7087–7111. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics.

Purdy, C.; Wang, X.; He, L.; and Riedl, M. 2018. Predicting generated story quality with quantitative measures. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 14, 95–101.

Ramsay, S. 2011. *Reading Machines: Toward and Algorithmic Criticism*. University of Illinois Press.

Riedl, M. O.; and Bulitko, V. 2012. Interactive Narrative: An Intelligent Systems Approach. *AI Magazine*, 34(1): 67.

Riedl, M. O.; and Young, R. M. 2010. Narrative Planning: Balancing Plot and Character. *Journal of Artificial Intelligence Research*, 39: 217–268.

Roemmele, M. 2016. Writing stories with help from recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Roemmele, M.; Gordon, A. S.; and Swanson, R. 2017. Evaluating story generation systems using automated linguistic analyses. Technical report, University of Southern California.

Sato, M. 2023. CNET Is Doing Big Layoffs Just Weeks after AI-generated Stories Came to Light. https://www.theverge.com/2023/3/2/23622231/cnet-layoffs-ai-articles-seo-red-ventures. Accessed: 2023-08-15.

Sawicki, P.; Grzes, M.; Goes, F.; Brown, D.; Peeperkorn, M.; and Khatun, A. 2023. Bits of grass: does GPT already know how to write like Whitman? In *14th International Conference for Computational Creativity*.

Sil, A.; and Yates, A. 2011. Extracting STRIPS Representations of Actions and Events. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, 1–8. Hissar, Bulgaria: Association for Computational Linguistics.

Silver, T.; Dan, S.; Srinivas, K.; Tenenbaum, J. B.; Kaelbling, L. P.; and Katz, M. 2023. Generalized Planning in PDDL Domains with Pretrained Large Language Models. arxiv:2305.11014.

Simon, N.; and Muise, C. 2022. TattleTale: Storytelling with Planning and Large Language Models. In *ICAPS Workshop on Scheduling and Planning Applications*.

Smith, G.; and Whitehead, J. 2010. Analyzing the Expressive Range of a Level Generator. In *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, PCGames '10. New York, NY, USA: Association for Computing Machinery. ISBN 9781450300230.

Tang, C.; Zhang, Z.; Loakman, T.; Lin, C.; and Guerin, F. 2022. NGEP: A Graph-based Event Planning Framework for Story Generation. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 186–193. Online only: Association for Computational Linguistics.

Valmeekam, K.; Olmo, A.; Sreedharan, S.; and Kambhampati, S. 2022. Large Language Models Still Can't Plan (A Benchmark for LLMs on Planning and Reasoning about Change). In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

Ware, S. G.; and Siler, C. 2021. Sabre: A Narrative Planner Supporting Intention and Deep Theory of Mind. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 17(1): 99–106.

Ware, S. G.; and Young, R. M. 2014. Glaive: a state-space narrative planner supporting intentionality and conflict. In *Tenth Artificial Intelligence and Interactive Digital Entertainment Conference*.

Yao, L.; Peng, N.; Weischedel, R.; Knight, K.; Zhao, D.; and Yan, R. 2019. Plan-and-Write: Towards Better Automatic Storytelling. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01): 7378–7385.

Ye, A.; Cui, C. Z.; Shi, T.; and Riedl, M. 2023. Neural Story Planning. In *The AAAI-23 Workshop on Creative AI Across Modalities*.

Yordanova, K.; and Kirste, T. 2016. Learning Models of Human Behaviour from Textual Instructions. In *Proceedings of the 8th International Conference on Agents and Artificial Intelligence*, ICAART 2016, 415–422. Setubal, PRT: SCITEPRESS - Science and Technology Publications, Lda. ISBN 9789897581724.

Zhang, Z.; Zhang, A.; Li, M.; and Smola, A. 2023. Automatic Chain of Thought Prompting in Large Language Models. In *The Eleventh International Conference on Learning Representations*.