TattleTale: Storytelling with Planning and Large Language Models

Nisha Simon, Christian Muise
Queen's University, School of Computing
Goodwin Hall, Kingston ON
20nis@queensu.ca, christian.muise@queensu.ca

Abstract

We explore how automated planning can be applied to Natural Language text generation in order to create narratives (stories) that are coherent and believable. While Large Language Models (LLMs) such as *GPT-3* can be used for narrative generation based on given input prompts, they lack coherence and can be prone to repetition and stilted language. We demonstrate the use of a planning model that provides scaffolding to an LLM so that its language generation is context-dependent in order to create more coherent and believable stories in a variety of domains. After manually extracting characters, objects, and locations from the story source, we create domain and problem encoding that captures the mechanics of the story. The output of a planner, taken one action at a time, is fed to the LLM to generate a narrative. We find that almost all nouns (characters, objects, and locations) and verbs (actions) of the plan are reflected in the generated story, and the resulting narrative is more coherent than stories that are generated using only plain text prompts to the LLM. Finally, gathering, curating, and modelling the source stories in PDDL is an additional contribution of our work that will be released publicly. Our work represents a key first step towards the novel application of planning technology to a neuro-symbolic approach for effective story generation.

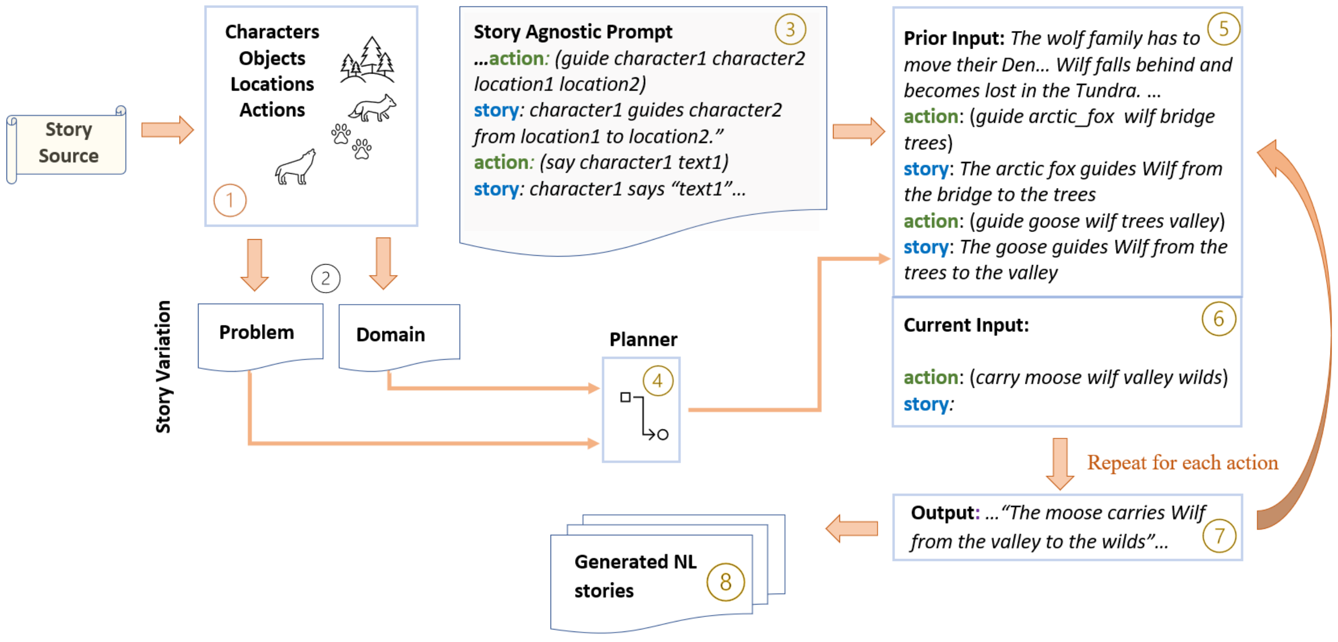
1 Introduction

In many walks of life we have increasingly come to rely on intelligent systems that automate both routine and high-level tasks for us in order to meet our desired goals, but we still know relatively little about how those systems really make their decisions and plan sequences of actions that fulfill those goals. Story telling or narrative generation can be used to determine if a particular intelligent industrial, financial or medical system truly understands relevant real-world concepts, or in explainable AI, to allow a system to explain the reasoning behind its selected actions to a human user. The system can generate language to articulate why it chose certain actions over others. In the future, these detailed explanations can then be used to train systems to pick the correct sequence of actions for a given related scenario. The domains for such applications can range from mechanical equipment repair, medical diagnosis assistance, to automatic vehicle navigation. Story telling can also be used for educational purposes, and in entertainment fields such as creating '*choose-your-own-adventure*' games (Riedl 2021). Story telling or narrative generation has various entertainment applications (Riedl 2016) such as text based games, and it can involve extending narratives to create new plots. Story telling has been used in several ways, for instance to create variations on a base story (Hayton et al. 2020) or to extend an existing story (Porteous et al. 2020). In this work, we propose to show how automated planning can be applied to Natural Language text generation in order to create believable and coherent narratives (stories) based on the given application. Although LLMs such as GPT-3 can be used for narrative generation based on given input prompts such as the first two or three lines of a narrative, they lack coherence, and can be prone to repetition and stilted language (Olmo, Sreedharan, and Kambhampati 2021; Castricato et al. 2021). We therefore accomplish our goal of creating coherent narratives in a variety of domains by building a model that will provide inputs to the Large Language Model that are more context-dependent, and that incorporate commonsense knowledge.

Creating a plan in storytelling involves creating a sequence of steps that are essential in reaching an end goal. For instance, the desired result of a fairy tale may be that a princess escapes from a tower, and the steps of the plan may involve the princess finding all the ingredients to create a magic spell that allows her to do so. The required characteristics of the plan may be *coherence*, that is to say, the steps of the story follow a logical outline, and *consistency*, which means that the characters act in predictable ways according to their individual traits and abilities and the parameters of the story (Hayton 2019).

Plans may be created in order to generate natural language utterances, a process that is also called Natural Language Generation (NLG) (Appelt 1982; Koller and Hoffmann 2010). Automated Planning can be applied to Natural Language text generation in order to create believable, coherent and engaging narratives (stories) that inform, entertain or educate the user based on the given application, in a variety of domains.

Large Language Models (LLMs) are a class of deep learning architectures that have been trained on a large amount of textual data. They can then be used to generate text based on given input prompts, and they do so one token (i.e., word) at a time. The LLM used in this paper is *GPT-J-6B* which is '*a 6 billion parameter, autoregressive text gener-*



Story Source
Characters
Objects
Locations
Actions
1
2
Story Variation
Problem
Domain
Story Agnostic Prompt
3
...action: (guide character1 character2 location1 location2)
story: character1 guides character2 from location1 to location2."
action: (say character1 text1)
story: character1 says "text1"...
Planner
4
Prior Input: The wolf family has to move their Den... Wilf falls behind and becomes lost in the Tundra. ...
5
action: (guide arctic_fox wilf bridge trees)
story: The arctic fox guides Wilf from the bridge to the trees
action: (guide goose wilf trees valley)
story: The goose guides Wilf from the trees to the valley
Current Input:
6
action: (carry moose wilf valley wilds)
story:
Repeat for each action
Output: ..."The moose carries Wilf from the valley to the wilds"...
7
Generated NL stories
8

Figure 1: System architecture showing the various components that are used to generate the natural language story. From the original story source, Step 1: characters, objects, locations, and actions are manually extracted. Step 2: *Problem* (story variation) and *Domain* (story mechanics) files are created. Step 3: Story agnostic prompts are created by hand as the initial input to the LLM in order to provide background and style information. Step 4: The planner creates a valid plan. Step 5: prior inputs, along with story agnostic prompts, and Step 6: the actions of the plan (Current input) which is the current action that is being processed, are iteratively used as input prompts into the LLM, to Step 7: generate a natural language story. The resulting output sentences are then collected together and compiled into a plain text file in Step 8 to form the complete generated story.

ation model' (Wang 2021) that was trained on a large combined dataset called '*The Pile*', that includes smaller datasets from sources such as PubMed Central, GitHub, Stack Exchange, and BookCorpus2 (Gao et al. 2020). In this study, we use classical planning with STRIPS and typing to create the plans. Some of the Natural Language metrics that are used in our research are based on POS (Part of Speech) tags identified in the stories, which are the grammatical classifications that indicate the function that a particular word performs in a sentence.

The mechanics of how LLMs work and the precise notation for the planning models are both standard in our approach. For further details, we refer the reader to general resources on each topic respectively (Brown et al. 2020; Haslum et al. 2019).

2 Generating Stories with Automated Planning and LLMs

The first step in our study was to obtain suitable datasets. Our initial selected datasets consist of children's stories, as these are likely to have simple vocabulary and basic grammar. The second step was to manually translate stories to a domain and problem encoding in the form of PDDL. The selected stories were manually anlaysed to extract the character names, locations, significant objects, and goals. An example of this is illustrated in Table 1. These were then used to create predicates and actions for encoding them in PDDL. The translation from story to PDDL is done by hand, by manually extracting characters, locations, objects, states, and goals. While the NLP technique of Named Entity Recognition (NER) could potentially be used to automatically extract the characters, locations and objects, the use of this technique may suffer from inaccuracies, as all entities may not be correctly recognized. In addition, if POS tagging is used to identify nouns and verbs this could also result in inaccuracies due to ambiguity. A word such as 'fire' or 'plan', for instance, may be a verb in one context but a noun in another.

A typical plan is about 10 to 20 actions long, and a generated story is of the corresponding length. The predicate and action names are designed under the constraints of PDDL syntax and some features such as the concept of two locations being connected, and therefore allowing a character to move between them, or a character being available to perform a certain action in one state but not in another, are shown by predicate names such as "isconnected" or "hasfriend". This step resulted in the creation of the '*Domain*' file. We also constructed the initial/original '*Problem*' file that was based on the original story. A snippet of the PDDL code of the Domain file for the '*The Way Home for Wolf*' story is shown in Figure 2. The *domain* file and the *problem* file were then sent as inputs to a classical planner, which generated a valid plan for the story. A generated plan for the '*The Way Home for Wolf*' story is shown in Listing 2.

Category	List of Corresponding Items
Characters	Wilf, polar bear, walrus, narwhal musk ox, arctic fox, goose, moose, bear moth, wolf family
Locations	iceberg, shore, ridge. trees, bridge, valley, wilds, stream, den
Actions	guide, carry
Objects	–
Initial State	Wilf is lost in the Tundra
Goal State	Wilf is safely back at the Den

Table 1: Examples of characters, actions and locations that were manually extracted by hand from the '*The Way Home for Wolf*' story. Note that this particular story has no explicit entities that are characterised as objects, whereas in stories like '*Robin Hood and the Golden Arrow*' we have objects like 'arrow' and 'target'.

We assume that the original story corresponds to a valid plan i.e. the character goals and author goals are achieved. However this may not always be the plan that is found. We also created variations of the story in additional problem instances. We ensure that the above domain and problem files result in valid plans that are found by the solver. After computing the plans, we fed the individual plan actions to the LLM and then obtained the corresponding next Natural Language line, one at a time as follows: First we include any background or real-world contextual information as the initial or '*hidden*' (i.e. not directly related to the specific actions in the story) input prompt (also called story-agnostic prompts) to the LLM . Second, we took one line at a time, in sequence, of the solved plan. Third, we appended this line to the hidden prompt for the LLM input. Fourth, we let the LLM generate one line of output (ignoring any other lines that were generated). Fifth, taking this one line of generated output, we added it to the end of the existing input prompt and used the result as the new input prompt. This process was repeated until all action lines of the plan had been used. Lastly, we then compiled the generated Natural Language lines into stories. The input prompt that was used, in addition to the final output of the LLM, was pasted into a text file as a complete story. The details of this process are illustrated in Figure 1.

Listing 1: Example of a story agnostic prompt for '*The Way Home for Wolf*' story

```
Input:

The wolf family has to move their Den. Wilf
    is a small wolf pup. The wolf family
    starts their journey to their new den,
    but Wilf is too small to keep up. Wilf
    falls behind and becomes lost in the
    Tundra.

Output:

He needs to get back to his family as soon
    as possible. Wilf has to go on an
    adventure to find his way home. Wilf is
    an adventure story. The story is about
    the journey of a wolf pup, to find his
    way home. This book is for children aged
     6-9 years. It is also suitable for
    those who are looking for a bit of
    adventure in their lives.
```

The story agnostic prompts provide background or common-sense information that guides the rest of the output but can be common to multiple stories e.g., for the 'Wilf' story, one of the story-agnostic prompts might be: 'Wilf is an animal, and he is a small wolf cub'. This can be extended to a story about another character for example, 'James is an animal, and he is a small hedgehog'. The story agnostic prompts are also useful for style purposes such as how the story is written, etc. The LLM requires at least two (or more) initial patterns of 'action' and 'story' prompts in order to recognize and generate the text. The 'pattern' is the combination of the story-agnostic prompts, as well as the initial inputs.

The example of Listing 1 shows that the LLM is initially given an input prompt consisting of a short paragraph of background information. In response, the LLM then generates some output of its own in the form of another short paragraph. The LLM can be considered to be a 'few-shot' learner, since with the story agnostic prompts and about two or three input prompts, it is able to generate valid output.

It should be noted that Figure 1 includes the precise input to the LLM. That is, we seed it with PDDL-formatted actions from the plan along with the story snippets, and it then generates the next story snippet for us. On the right of Figure 1, aside from 'Prior Input:' and 'Current Input:', those complete strings are fed as written to the LLM, and the 'Output:' box lists the precise response from the LLM. Thus, the LLM receives inputs in the exact form shown in Listing 1. The generated output for the current iteration is then added to the initial prompts along with the previous outputs and then fed back into the LLM. This process continues until all the actions in the plan have been fed into the LLM, with the input growing in size each time. Note that although the input size is steadily increasing each time a new action is added, only the first generated line of any output that is produced by the LLM in the current iteration is used. This is due to the fact that using more than one line of the LLM's output from the current iteration will likely introduce incoherent text into future inputs. Therefore, an interesting effect of this character of the LLM is that the LLM cannot simply 'fill up space' in the story while simultaneously remaining faithful to the plot, nor is it likely to be able to generate a coherent story if some of the specified actions are removed from the plan.

2.1 Dataset Curation

LLMs can sometimes display skewed results in their produced output, based on the inputs with which they are first seeded. Therefore, gathering, modeling, and curating the datasets is an important contribution. To ensure diversity and richness in the prompts, the input stories were drawn from a wide variety of sources such as folktales from Asia, First Nations stories, as well as North American and European fables. We curated the prompts to prevent bias due to gender, race, ethnicity, or geographic region. Examples of stories that were used are children's stories such as '*The Way Home for Wolf*' (Bright and Field 2020), '*Robin Hood and the Golden Arrow*' (San Souci and Lewis 2010), and '*The Paper Bag Princess*' (Munsch and Martchenko 1980). The input stories were selected so as to contain simple language (vocabulary) and sentence construction.

3 Implementation and Evaluation

The preliminary goal of the research was to synthesise a model with the intention of generating plans that correspond to given story lines. The initial step was therefore the development of a planning model to create the story skeleton. The variations on the base story (story source) were generated based on prompts that had been fed into the LLM as inputs. For the purposes of training, the initial prompts were based on children's stories that contain simple sentences and basic grammatical constructs. The main objective was to generate the skeleton story, feed the prompts to the LLM incrementally, and use the resulting output as new inputs to the system. A snippet of the PDDL code of the Domain file for the '*The Way Home for Wolf'* story is shown in Figure 2. This can be considered part of the 'abstract' version of the PDDL model, while the output components generated by the planner in Table 2, can be considered part of an 'instance' of the model.

Examples of a predicate (items that can be either TRUE or FALSE in the domain) from the story '*Robin Hood and the Golden Arrow*' (San Souci and Lewis 2010) are that Robin Hood is an archer and that he is an outlaw. As an example of an action, we consider the action *'hide-character'* as shown in Figure 5, which is one particular action that is used in the plan that generates this story.

The parameters of the action in Figure 5 are an '*archer*', an '*event*' and a '*disguise*', which are instantiated as '*Robin Hood*', the '*archery contest*', and a '*beggar*' respectively. The preconditions are that the archer learns of the event and the archer is an outlaw. A character who is an outlaw requires a disguise in order to prevent themselves from being captured while attending the event. Once all preconditions are met, the effect of the action is that the archer is able to

```
(:predicates

   (hasFriend ?l1 - location ?f - friend)
   (isConnected ?l1 ?l2 - location)
   (at ?l1 - location ?w1 -wilf)
   (canGuide ?l1 ?l2 - location
             ?f - friend)
   (canCarry ?l1 ?l2 - location
             ?f - friend))
```

Figure 2: Predicate portion of domain file PDDL code for '*The Way Home for Wolf*' story. This is part of the actual PDDL code which can be considered the 'abstract' version, as opposed to an 'instance'.

```
(:action guide

   :parameters
      (?f - friend ?w - wilf
       ?l1 ?l2 - location)

   :precondition
      (and
         (isConnected ?l1 ?l2)
         (at ?l1 ?w)
         (hasFriend ?l1 ?f)
         (canGuide ?l1 ?l2 ?f))

  :effect
      (and
         (at ?l2 ?w )
         (not(at ?l1 ?w))
         (not(hasFriend ?l1 ?f))))
```

Figure 3: Actions portion of domain file PDDL code for '*The Way Home for Wolf*' story. The action '*guide*' is shown. This is part of the actual PDDL code which can be considered the 'abstract' version, as opposed to an 'instance'.

```
(:action carry

   :parameters
      (?f - friend ?w - wilf
       ?l1 ?l2 - location)

   :precondition
      (and
         (isConnected ?l1 ?l2)
         (at ?l1 ?w)
         (hasFriend ?l1 ?f)
         (canCarry ?l1 ?l2 ?f))

   :effect
      (and
         (at ?l2 ?w )
         (not(at ?l1 ?w))
         (not(hasFriend ?l1 ?f))))
```

Figure 4: Actions portion of domain file PDDL code for '*The Way Home for Wolf*' story. The action '*carry*' is shown. This is part of the actual PDDL code which can be considered the 'abstract' version, as opposed to an 'instance'.

```
(:action hide_character

    :parameters (?a1 - archer
                 ?d - disguise
                 ?e - event)

    :precondition (and
                   (learn ?a1  ?e)
                   (isOutlaw ?a1 ))

    :effect (disguised_As  ?a1 ?d)
)
```

Figure 5: An action example in PDDL language - '*hide-character*', from the '*Robin Hood and the Golden Arrow*' story.

Listing 2: Example of a generated plan for '*The Way Home for Wolf*' story

```
(carry polar_bear wilf iceberg shore help_is_here)
(carry walrus wilf shore ridge to_the_ridge)
(carry musk_ox wilf ridge bridge hold_on)
(guide arctic_fox wilf bridge trees over_the_bridge)
(guide goose wilf trees valley though_the_trees)
(carry moose wilf valley wilds stay_with_me)
(guide bear_moth wilf wilds stream be_brave)
(carry wolf_family wilf stream den welcome_back)
```

Item Type	**List of Corresponding Items**
Action: parameters	**carry :** polar bear, Wilf, Iceberg, shore, 'Help is here'
Preconditions	hasFriend (iceberg, polar bear), isConnected (iceberg, shore), at iceberg (Wilf), canCarry (iceberg, shore, polar bear), canSay ('Help is here')
Effects	at shore (Wilf), not at iceberg (Wilf), not hasFriend (iceberg, polar bear)

Table 2: Portion of an output plan component showing how a particular action, which in this case is '*carry*' is (a) built up based on the *preconditions* that are required for that action to occur and (b) results in *effects* caused by that action. This is an example of an 'instance', while the actual PDDL code is the 'abstract' version.

disguise themselves and therefore they can participate in the event.

Creating the appropriate context is a key element of creating believable stories. The plan that leads to the generated story required a method of determining which characters, objects, and actions are appropriate for the particular story. The variations of the story need to be believable and interesting while still following the skeleton in their salient points. For instance, a story about genies and magic spells that takes place many years ago might become jarring if the story variation were suddenly to include futuristic spaceships or time-travelling robots. Reliance on the symbolic planning model was vital in maintaining this type of coherence. This is also a critical purpose of using appropriate story-agnostic prompts to 'set the scene' for the generated story.

There are two main goals to be considered when creating a story: the *author* or overarching goal (Lebowitz 1985) that brings the story to a successful conclusion, and the multi-agent or *character* goals. Characters in the story must work towards achieving the main author goal while staying true to their own beliefs and characteristics; that is to say, they must be consistent in their actions.

An example of an encoded action that forms part of a valid plan for a given domain and problem file is given in the snippet in Table 2. We see that the Polar Bear fulfills his character goal by being helpful and carrying Wilf to the next leg of the young wolf cub's journey. In addition, Wilf fulfills his character goal of being an adventurous explorer and also works towards achieving the author goal of being reunited with his family.

It should be noted that if the input plan contains several different viable paths leading to the end goal i.e., if there are multiple ways to reach the conclusion or main aim of the story, then the results will be several variations of the story each time the LLM is queried. Therefore, although a single version of the example story is presented in the paper, a plan that contains multiple paths is capable of creating several variations of a story if the LLM is repeatedly queried. For example, in the '*The Way Home for Wolf*' story, the main goal is for Wilf to find his way back home to the Wolf den. If there are multiple animal friends that can guide Wilf from location 1 to location 2 on the way back to the den, then one version of the story may generate text that shows friend 1 being a guide, while querying the LLM again may generate text that shows friend 2 being a guide instead. These variations can be achieved by appropriate additions to the Domain and Problem files. For instance, if the underlying PDDL is constructed so that both the polar bear and the narwhal are able to carry Wilf from the iceberg to the shore, then the planner will automatically pick either one animal friend or the other to get Wilf to the shore each time the plan is generated. In Listing 2 we see that the planner has chosen the polar bear to carry Wilf from the iceberg to the shore, but it could just as easily have created an alternative plan where it falls to the narwhal to perform this act, with the remainder of the original plan being concatenated and then continuing as before to achieve the same objective.

The planner used was the online solver[1] (Muise 2016). The LLM model used was GPT-J-6B eleuther (Wang 2021) that is available for limited public trial online through a browser interface. The evaluation metrics used in this study include part-of-speech tags of nouns and verbs to indicate how many of the characters, objects, locations (nouns), and actions (verbs) from the plan are reflected in the output of the LLM. That is to say, we evaluate how well the generated story from the LLM mirrors the output of the planner. The generated stories are also judged on whether or not they achieved their required author and character goals, as well as on coherence. Castricato et al. define coherence as '*any perceivable relationship between events in a story*' (Castricato et al. 2021).

4 Results

The output of the LLM shows that when its inputs are provided as actions from generated plans, both the author and character goals are achieved. For instance, in the story '*The Way Home for Wolf*' (Bright and Field 2020), the title character, Wilf, fulfills the author goal by successfully finding his way home. The characters of the Moose, Goose, Musk Ox, Arctic Fox, and Wilf's other animal friends fulfill their character goals by being helpful and friendly and guiding Wilf back to his den. In Figure 3 we see that some animal friends *guide* Wilf from one location to another while others *carry* Wilf from one location to another. This is because only the comparatively larger animals such as the walrus, musk ox and moose are strong enough to carry Wilf, but the other smaller animals like the tiny arctic fox and the diminutive bear moth have to rely on their wits, rather than on brute strength, to guide little Wilf on his journey home.

[1]https://solver.planning.domains/

Listing 3: Example of a natural language story that has been generated by the LLM based on input actions from a valid plan for '*The Way Home for Wolf*' story

```
The Polar Bear carries Wilf from the iceberg
     to the shore and says "Help is here"

The Walrus carries Wilf from the shore to
    the ridge and says "To the Ridge"

he Musk Ox carries Wilf from the ridge to
    the bridge and says "Hold on"

The Arctic Fox guides Wilf from the bridge
    to the trees and says "Over the bridge"

The Goose guides Wilf from the trees to the
    valley and says "Through the trees"

The Moose carries Wilf from the valley to
    the wilds and says "Stay with me"

The Bear  guides Wilf from the wilds to the
    stream and says "Be Brave"

The Wolf  carries Wilf from the stream to
    the den and says "Welcome back"
```

Another example is the story '*Robin Hood and the Golden Arrow*' (San Souci and Lewis 2010) where Robin Hood fulfills both the character goal of being a fabled archer and also the author goals of entering the archery contest in disguise to trick the Sheriff, winning the archery competition and then escaping from the trap that was set for him by the wicked Sheriff of Nottingham.

The Part of Speech (POS) tags are codes that represent the grammatical function that is performed by a particular word in a sentence, for example 'noun' or 'verb'. The verbs '*carry*' and '*guide*' for instance are successfully included in the generated story, as are the character names (nouns) such as *Goose, Musk Ox*, and so on, of Wilf and his animal friends.

It should be noted that the text of the LLM output should be considered the 'average' or representative output, since the generated output of the LLM may vary slightly every time the process is repeated, especially for more complex stories like '*Robin Hood and the Golden Arrow*' and '*The Paper Bag Princess*. Although the gist and the main thread of the story are maintained over several iterations, exactly the same wording may not be generated each time.

Listing 4: Example of a natural language story that has been generated by the LLM based on input actions from a valid plan for the '*Robin Hood and the Golden Arrow*' story

```
The sheriff announces a archery contest.

Robin Hood learns about the archery contest.

Robin Hood disguises himself as a beggar to
    enter the archery contest.

Robin Hood participates in the archery
    contest

The arrow hits the target at the center.

Robin Hood wins the golden arrow.
```

As can be seen from the POS counts in Table 3 and the output snippets in Listing 3 and Listing 4 the LLM output captures almost all the nouns and verbs from the PDDL plan. It is interesting to note that while the LLM captures the gist of the plan, certain nouns and verbs are sometimes added or substituted for clarity. For example, the verb '*say*' is added to the '*The Way Home for Wolf*' story for clarity even though this particular verb is implied but not explicitly used in the found plan. The verb '*hits*' is added to the generated '*Robin Hood and the Golden Arrow*' story, although this word is not explicitly used in the plan. Also, it is interesting to note that the LLM does not always recognize unfamiliar nouns. For example, '*Bear Moth*' is replaced with '*Bear*' and '*Wolf Family*' is replaced with '*Wolf*' in the generated story. It is possible that the LLM needs more fine-grained instruction in the form of hidden prompts on how best to handle and recognize unfamiliar compound nouns.

The generated natural language stories also display the required qualities of coherence and consistency. The output was considered to be 'coherent' if there were no grammatical or logic errors e.g., verbs and nouns were in agreement, and a character could not be in two locations at the same time. An example of incoherent text is provided in the Listing 6 where the LLM is allowed free rein and therefore devolves into repetition. Our generated outputs in Listing 4 and Listing 3 stand in contrast to the output shown in Listing 6. The output shown in Listing 6 indicates that in the absence of relevant PDDL input prompts, the LLM loses the story thread and instead resorts to basic repetition with no regard for the plot outline or for the prior actions that have already occurred in the story.

We used quantitative metrics in the paper, but we also intend to use more qualitative metrics by distributing our generated stories to human participants in a survey, and then asking them to provide feedback by judging the coherence and believability of the generated stories. More thorough empirical evaluation with human participants is currently beyond the scope of this paper.

Listing 5: Story that has been generated by the LLM based on input actions from a valid plan for the '*Paper Bag Princess*' story

```
Princess Elizabeth is a beautiful princess
    who lives in a magnificent castle.
    Princess Elizabeth is engaged to marry
    Prince Ronald.
The Dragon attacks and destroys the castle
    and Princess Elizabeth's clothes and the
     Dragon kidnaps Prince Ronald.
Princess Elizabeth wears a paper bag because
     her clothes are destroyed, and she
    follows the Dragon.
Princess Elizabeth chases the Dragon, and it
     is charmed by Princess Elizabeth.
Princess Elizabeth flatters the Dragon, and
    the Dragon likes Princess Elizabeth.
The Dragon breathes large flames.
The Dragon flies fast.
The Dragon falls asleep.
Princess Elizabeth saves Prince Ronald from
    the Dragon.
Prince Ronald insults Princess Elizabeth,
    and she wears a paper bag because she is
     wearing no clothes.
Princess Elizabeth calls off the wedding,
    and she is wearing a paper bag.
```

5 Related Work

In numerous domains and applications, it becomes necessary to convert plans to Natural Language instructions and vice versa (Hayton et al. 2020; Miglani and Yorke-Smith 2020; Steinert and Meneguzzi 2020; Feng, Zhuo, and Kambhampati 2018). Natural language instructions can also be created from given plans in order to make the plan more understandable to a human user. Automated Planning can therefore be used for story telling, and text generation can be viewed as a planning task (Koller and Petrick 2011).

Efforts to first plan out skeletons of stories include Yao et al. who have used planning as a neural generation 'plan-and-write' framework (Yao et al. 2019) to build a system that performs open-domain story generation, which is to generate a story based on a given title as well as a plan. Their system alternates between planning the steps of the story and then writing it. However their stories are limited in length (five sentences) and vocabulary size (33,215), and they rely on story lines that are extracted from existing story corpora (Yao et al. 2019). Unlike *TattleTale*, this system does not use a PDDL planner, and in contrast, our *TattleTale* system has the potential to work without the input of whole original stories.

Several story-telling approaches already exist in the literature. For example, interactive narrative planning is used in the entertainment field (Porteous et al. 2021) where an extended narrative or a range of plans are created based on a baseline story. Storytelling with multiple agents has been studied in systems such as *SABRE* (Ware and Siler 2021).

Text generation or sentence generation is an integral aspect of storytelling. It can be performed by isolating relevant

Story	POS tag	Plan	LLM story
The Way Home for Wolf	Noun	18	18
	Verb	2	3
Robin Hood and the Golden Arrow	Noun	8	7
	Verb	6	7
Paper Bag Princess	Noun	6	7
	Verb	10	19

Table 3: Number of nouns and verbs found in the PDDL plan that are successfully duplicated in the LLM output

Listing 6: Illustration of repetition in the unrestricted LLM output for '*The Way Home for Wolf*' story. When the PDDL input is not provided in the form of actions from the plan, the LLM defaults back to repeating the same text over and over.

```
Output:

But, Wilf has to travel home on his own. He
    has to travel through the Tundra. He
    needs to cross the river. He needs to
    cross the river. He needs to cross the
    river. He needs to cross the river. He
    needs to cross the river. He needs to
    cross the river. He needs to cross the
    river. He needs to cross the river. He
    needs to cross the river. He needs to
    cross the river. He needs to cross the
    river. He needs to cross the river. He
    needs to cross the river. He needs to
    cross the river. He needs to cross the
    river
```

actions and objects, using Part of Speech (POS) tags to identify verbs and nouns, and using a prior dictionary of terms. Sentence generation or utterance generation can be regarded as a planning problem (Koller and Stone 2007). Planning has also been used in this context for sentence generation. In addition to using the correct grammar, recognizing the appropriate context of a sentence is another challenge in sentence generation. Rule-based text generation models can be restrictive, and may not be able to encompass a wide array of domains. In contrast to the above, our work uses an LLM to generate the individual sentences corresponding to distinct actions in the plan.

6 Conclusion

In this study we have shown how Automated Planning can be applied to Natural Language text generation in order to create narratives (stories) that are coherent and believable. In our *Tattletale* system, we have overcome the issues of a lack coherence, repetition, and stilted language to which LLMs may be prone. We accomplish this by demonstrating the use of a planning model that provides scaffolding to the LLM so that its language generation is context-dependent in order to create more coherent and believable stories in a variety of domains. Plans are vital to the construction of the stories since without the plan, the output text loses coherence and the LLM could not create a valid story. We find that almost all of the nouns (characters, objects, and locations) and verbs (actions) of the plan are reflected in the generated story, and that the resulting narrative is more coherent than stories that are generated using only plain text prompts to the LLM. Finally, gathering, curating, and modeling the source stories in PDDL was an additional contribution of our work. Our research opens the door to many planning-oriented extensions. Future work will include Epistemic planning giving more believable agent understanding, including the goals of deception/misconception in epistemic planning to tell better stories, and the inclusion of linear temporal logic constraints to direct how a story might unfold.

Ethical/Societal Impact

We also consider the potential broader impact of our work, including its ethical aspects and future societal consequences. As indicated in the Dataset Curation section 2.1, if inputs to LLMs are not appropriately directed, LLMs can sometimes display bias in their produced output, digress into irrelevance (as shown in Listing 6), as well as generate harmful, potentially toxic or discriminatory outputs. We avoid these issues by strictly controlling the prompts that are given to the LLM. Prompts were therefore curated to prevent bias due to gender, race, ethnicity, or geographic region and also to remove any potentially negative inputs. To ensure diversity and richness in the prompts, the input stories were drawn from a wide variety of sources such as folktales from Asia, First Nations stories, as well as North American and European fables.

References

Appelt, D. E. 1982. Planning Natural-Language Utterances. In *AAAI*, 59–62.

Bright, R.; and Field, J. 2020. *The Way Home for Wolf*, volume 1. Scholastic Press.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Castricato, L.; Frazier, S.; Balloch, J.; Tarakad, N.; and Riedl, M. 2021. Automated Story Generation as Question-Answering. *arXiv preprint arXiv:2112.03808v1*.

Feng, W.; Zhuo, H. H.; and Kambhampati, S. 2018. Extracting action sequences from texts based on deep reinforcement learning. *arXiv preprint arXiv:1803.02632*.

Gao, L.; Biderman, S.; Black, S.; Golding, L.; Hoppe, T.; Foster, C.; Phang, J.; He, H.; Thite, A.; Nabeshima, N.; Presser, S.; and Leahy, C. 2020. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *arXiv preprint arXiv:2101.00027*.

Haslum, P.; Lipovetzky, N.; Magazzeni, D.; and Muise, C. 2019. *An Introduction to the Planning Domain Definition Language*. Morgan & Claypool. ISBN 9781627058759.

Hayton, T. 2019. *Acquiring Planning Models from Narrative Synopses*. Ph.D. thesis, Teesside University.

Hayton, T.; Porteous, J.; Ferreira, J. F.; and Lindsay, A. 2020. Narrative Planning Model Acquisition from Text Summaries and Descriptions. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI-20)*.

Koller, A.; and Hoffmann, J. 2010. Waking Up a Sleeping Rabbit: On Natural-Language Sentence Generation with FF. In *Proceedings of the Twentieth International Conference on Automated Planning and Scheduling (ICAPS 2010)*.

Koller, A.; and Petrick, R. P. 2011. Experiences with planning for natural language generation. *Computational Intelligence*, 27(1): 23–40.

Koller, A.; and Stone, M. 2007. Sentence generation as a planning problem. Technical report, Columbia University.

Lebowitz, M. 1985. Story-telling as Planning and Learning. In *Poetics*.

Miglani, S.; and Yorke-Smith, N. 2020. NLtoPDDL: One-Shot Learning of PDDL Models from Natural Language Process Manuals. In *ICAPS'20 Workshop on Knowledge Engineering for Planning and Scheduling (KEPS'20). ICAPS*.

Muise, C. 2016. Planning.Domains. In *The 26th International Conference on Automated Planning and Scheduling - Demonstrations*.

Munsch, R.; and Martchenko, M. 1980. *The Paper Bag Princess*, volume 1. Annick Press.

Olmo, A.; Sreedharan, S.; and Kambhampati, S. 2021. GPT3-to-plan: Extracting plans from text using GPT-3. *arXiv preprint arXiv:2106.07131*.

Porteous, J.; Ferreira, J. F.; Lindsay, A.; and Cavazza, M. 2020. Extending Narrative Planning Domains with Linguistic Resources. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020)*.

Porteous, J.; Ferreira, J. F.; Lindsay, A.; and Cavazza, M. 2021. Automated narrative planning model extension. *Autonomous Agents and Multi-Agent Systems*, 35(2): 1–29.

Riedl, M. 2021. An Introduction to AI Story Generation. *The Gradient*.

Riedl, M. O. 2016. Computational narrative intelligence: A human-centered goal for artificial intelligence. *arXiv preprint arXiv:1602.06484*.

San Souci, R. D.; and Lewis, E. 2010. *Robin Hood and the Golden Arrow*, volume 1. Scholastic Press.

Steinert, M.; and Meneguzzi, F. 2020. Planning Domain Generation from Natural Language Step-by-Step Instructions. In *Proceedings of the Twentieth International Conference on Automated Planning and Scheduling (ICAPS 2020)*.

Wang, B. 2021. Mesh-Transformer-JAX: Model-Parallel Implementation of Transformer Language Model with JAX. https://github.com/kingoflolz/mesh-transformer-jax.

Ware, S. G.; and Siler, C. 2021. The Sabre Narrative Planner: Multi-Agent Coordination with Intentions and Beliefs. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 1698–1700.

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.