

---

# Interleaving a Symbolic Story Generator with a Neural Network-Based Large Language Model

---

**Jingwen Xiang**

JXIANG@SMITH.EDU

**Zoie Zhao**

ZZHAO39@SMITH.EDU

**Mackie Zhou**

MZHOU@SMITH.EDU

**Megan McKenzie**

MMCKENZIE@SMITH.EDU

**Alexis Kilayko**

AKILAYKO@SMITH.EDU

**Jamie C. Macbeth**

JMACBETH@SMITH.EDU

Department of Computer Science, Smith College, Northampton, MA 01063 USA

**Scott Carter**

SCOTT.CARTER@TRI.GLOBAL

**Katharine Sieck**

KATE.SIECK@TRI.GLOBAL

**Matthew Klenk**

MATT.KLENK@TRI.GLOBAL

Toyota Research Institute, 4440 El Camino Real, Los Altos, CA 94022, USA

## Abstract

Research in deep learning has recently produced models of natural language that are capable of generating natural language output which, at a glance, has strong similarities to that written by intelligent humans. However, the texts produced by deep learning-based large language models (LLMs), upon deeper examination, reveal the challenges that they have in producing outputs that maintain logical coherence. One specific application area of interest for LLMs is in fictional narrative generation, a mode of operation in which stories are generated by the model in response to a prompt text that indicates the start of a story or the desired style of writing to be produced. In this paper we present an initial study into a method for combining an LLM with a symbolic system to perform story generation. The method generates stories by repeatedly prompting the LLM by interleaving the output of a state-of-the-art LLM with the output of a classic story generation system, while attempting to control and shape the output of the LLM. We present a number of stories generated with a prototype interleaving system, and discuss the qualities of the stories and challenges for future development of the method.

## 1. Introduction

Large language models (LLMs) have garnered increased attention recently due to their apparent ability to generate natural language output that closely resembles that written by people. But recent work has shown the limitations of LLMs. Many LLMs have restricted scope because they are trained to perform well on tasks similar to those that LLMs are already known to perform well, making it difficult for the models to develop new capabilities. Also, the data that LLMs are trained on are collected through human labeling that is not performed by experts or by the task authors. The

Prompt:	1. One day, Joe was thirsty. Joe wanted not to be thirsty. Joe wanted to be near the water. Joe went to the river. Joe was near the river. Joe drank the water. Joe was not thirsty. The end.  2.
GPT-3 response:	Joe is a person who needs water. Joe is near the river. Joe wants to be near the river. Joe wants to be near the water. Joe wants the water. Joe is thirsty. Joe is the person who is thirsty. Joe is the person who is not thirsty. Joe is the person who is near the river. Joe is the person who is near the water. Joe is the person who wants to be near the river. Joe is the person who wants to be near the water. Joe is the person who is near the river. Joe is the person who is near the water. Joe is the person who is thirsty. Joe is the person

Figure 1. A demonstration of prompting an LLM for story generation. The prompt provides a complete example story (in this case a simple TALE-SPIN story) and an enumeration pattern to indicate to the LLM (GPT-3) that it should compose a second story in the style of the first.

complexity of the data is limited due to the costs of collecting such data, resulting in easier tasks. These tasks may in turn contain issues with noise, correctness, and distribution that can reduce the interpretability of the resulting performance of the models (Srivastava et al., 2022).

One specific application area of interest for LLMs is in fictional narrative generation, a mode of operation in which stories are generated from the model in response to a prompt text that indicates the start of the story, or the desired style of writing to be produced. Here too, LLMs face significant challenges: a recent survey paper, reflecting on the performance of LLMs, states that “...generating coherent and logical story plots is still far from being solved” (Alabdulkarim et al., 2021). Specifically, LLMs are “...prone to degenerating into language models that pay little attention to the writing prompt” (Fan et al., 2018).

This paper presents a preliminary study into a method for keeping LLMs “on the rails” using a symbolic story generation system. The methods generate stories by mixing the output of a state-of-the-art LLM with the output of a classic story generation system, while attempting to control and shape the output of the LLM. The paper begins with brief descriptions of Generative Pre-trained Transformer 3 (GPT-3), the deep-learning based LLM used in our studies, and of TALE-SPIN, a predictable symbolic story generation system whose stories form the basis of our neural-symbolic mixing. We continue with a description of our novel mixed neural-symbolic methods for generating stories, which we call *interleaving*, and with examples of generated stories using these methods. We conclude with discussions of the method, the generated stories, and plans for future work.

## 2. GPT-3

GPT-3 (Generative Pre-trained Transformer) by OpenAI<sup>1</sup> is a third-generation deep learning neural network model capable of producing human-like text. The language model is trained on an unlabeled dataset consisting of masses of internet text, sourced from websites such as Wikipedia (Floridi & Chiriatti, 2020). GPT-3's 2019 predecessor, GPT-2, used 1.5B parameters trained on 40B tokens of internet text, while the 2020 model of GPT-3 uses 175B parameters trained on 499B tokens of web content, an increase which has seen the improvement in coherency of generated texts as compared to its predecessor (Dale, 2021).

GPT-3 takes in a small amount of user input text called the *prompt* and generates large amounts of relevant, proper text considered a statistically good fit, closely mimicking human-generated texts as a result. In addition to providing a prompt, parameters may be optionally adjusted to affect the output text. Figure 1 provides an example of typical prompting strategies for generating stories using an LLM, and how this prompting strategy frequently fails to generate stories with plot development.

## 3. Tale-Spin

TALE-SPIN is a classical symbolic story generator system that produces Aesop's Fables-like stories in English. A TALE-SPIN story focuses on the activities of a main character and a single main problem that they are motivated to solve. To make the plots of stories interesting, various obstacles are introduced that foil characters as they carry out their plans to achieve their goal of solving the main problem in the story. TALE-SPIN is described in Meehan (1977, 1981b) and in a recent survey paper on automatic story generation (Alhussain & Azmi, 2021). TALE-SPIN's output may be viewed broadly as a narration of generalized planning, goal-reasoning, and problem-solving processes. Although in this paper we refer to TALE-SPIN, we used stories generated by a freely available "micro" version of TALE-SPIN (Meehan, 1981a) which was significantly simpler than the original.

TALE-SPIN uses planning structures to organize knowledge to plan for characters' activities in a reasonable way and maintain story coherence. Planning structures are represented in Conceptual Dependency (Schank, 1972, 1975) as chains of causes and effects. TALE-SPIN incorporates a set of non-procedural structures which are divided into five classes: data about individual characters, data common to classes of characters, knowledge indicating how to achieve goals, memory, and default personal relationships. However, the world within the story is not fixed. New characters and objects can be introduced and personal relationships can change.

TALE-SPIN's story generator algorithm consists of three main components: a problem solver, which creates subgoals and actual events, an assertion mechanism, which adds an event to a "working memory", and an inference mechanism which generates the consequences of an event. When an event is asserted, it is added to memory and its consequences are computed and asserted, and so on. When a goal is asserted, it is not added to the memory, but instead, it triggers the problem solver. The resulting story structure is fed into a simple English generator for surface realization in natural language. Here is an example of a very simple TALE-SPIN story:

---

1. <https://openai.com/api/>

Prompt:	<p>1. One day, Joe was thirsty. Joe wanted not to be thirsty. Joe wanted to be near the water. Joe went to the river. Joe was near the river. Joe drank the water. Joe was not thirsty. The end.</p> <p>2. One day, Irving was hungry. Irving knew that Irving was hungry. Irving wanted not to be hungry.</p>
GPT-3 response:	<p>Irving wanted to be near the food. Irving went to the kitchen. Irving ate the food. Irving was not hungry. The end. 3. One day, Irving was in the jungle. Irving wanted not to be in the jungle. Irving wanted to be in the city. Irving went to the city. Irving was in the city. Irving was not in the jungle. The end. 4. One day, Irving was in the kitchen. Irving wanted not to be in the kitchen. Irving wanted to be in the bedroom. Irving went to the bedroom. Irving was in the bedroom. Irving was not in the kitchen. The end. 5. One day,</p>

Figure 2. An example of prompting an LLM for story generation completion of a TALE-SPIN story. The prompt uses a complete TALE-SPIN story to dictate style, and the beginning of a second TALE-SPIN story to provide guidance.

One day, Joe was thirsty. Joe wanted not to be thirsty. Joe wanted to be near the water. Joe went to the river. Joe was near the river. Joe drank the water. Joe was not thirsty. The end.

TALE-SPIN stories are a good choice for studies of neural-symbolic story generation because it is easy to generate TALE-SPIN stories with a range of simple or complex plots. Also, TALE-SPIN stories are expressed using simple sentences, and they have a lot of superfluous or redundant information. TALE-SPIN has the ability to generate uninteresting stories that are good candidates to be enhanced and made more interesting through completion by an LLM.

TALE-SPIN also typically generates a story “preamble” containing all of the “world facts” of the story premise, including the existence of inanimate objects (such as the location of water or food), story characters, their locations, beliefs, personalities, and social relationships. The story preamble for the previous story reads as follows:

Once upon a time ... Joe was near the cave. Joe knew that Joe was near the cave. Irving was near the oak-tree. Irving knew that Irving was near the oak-tree. Joe knew that Irving was near the oak-tree. The water was near the river. Joe knew that the water was near the river. The honey was near the elm-tree. Irving knew that the honey was near the elm-tree. The worm was near the ground. Joe knew that the worm was near the ground. Irving knew that Joe was near the cave. The fish was near the river. Irving knew that the fish was near the river.

We did not use TALE-SPIN story “preambles” in our studies of combining TALE-SPIN with an LLM, largely due to limitations on the allowed length of a prompt text in the LLM API. Figure 2

TALE-SPIN Story Segment	GPT-3 Prompt	GPT-3 Response Segment	Accumulated Story
Joe was thirsty.	Joe was thirsty.	Joe wanted to have a drink. (...Joe wanted to know where to find a drink. Joe thought that Joe did not know where to find a drink.)	Joe was thirsty. Joe wanted to have a drink
Joe was near the river.	Joe was thirsty. Joe wanted to have a drink. Joe was near the river.	Joe saw a cup. ( ... Joe took the cup. Joe found water. Joe drank the water.)	Joe was thirsty. Joe wanted to have a drink. Joe was near the river. Joe saw a cup.
Joe drank the water.	Joe was thirsty. Joe wanted to have a drink. Joe was near the river. Joe saw a cup. Joe drank the water.	Joe saw the cup and drank the water.	Joe was thirsty. Joe wanted to have a drink. Joe was near the river. Joe saw a cup. Joe drank the water. Joe saw the cup and drank the water.

Figure 3. An example of interleaving a TALE-SPIN story with GPT-3. In each iteration, the next TALE-SPIN story segment is appended to the accumulated story from the previous iteration, and is fed as a prompt to GPT-3. The TALE-SPIN story segment and the first sentence of the GPT-3 response are appended to create the new accumulated story for the current iteration. Additional text generated by GPT-3 after the first sentence was not used, but is shown in parentheses in the GPT-3 Response Segment column.

provides an example of prompting an LLM using a complete TALE-SPIN story to dictate style, and using the beginning of a second TALE-SPIN story to provide guidance.

#### 4. Interleaving with an LLM

Our novel interleaving process puts together story prompts written by TALE-SPIN and completions done by GPT-3. The process starts with feeding the first prompt—the first sentence of a story written by a TALE-SPIN—to GPT-3. Then, GPT-3 completes the story based on the first prompt. Depending on the parameters set for the completion and also the number of tokens provided, GPT-3’s completion is likely to be multiple sentences. However, the system parses the completion, keeps only the first sentence of the completion, and appends that sentence to the end of the first prompt.

The second prompt is created by appending the second sentence of the TALE-SPIN-written story to the end of the first sentence of GPT-3’s completion. Therefore, the second prompt is the

concatenation of the first sentence from the story, the first sentence from GPT-3’s completion, and the second sentence from the story. Again, the second prompt is fed to GPT-3, and the first sentence of GPT-3’s completion is taken out and appended to the end of the second prompt. The process continues by repeatedly appending one sentence from the TALE-SPIN story and one sentence from GPT-3’s completion to the existing prompt and feeding the updated prompt to GPT-3 each time. We stress that, in the studies presented herein, the TALE-SPIN stories are fixed and do not in any way see the output of the GPT-3 LLM or adapt to it.

Figure 3 illustrates an example of the prompting pattern we used for interleaving. Figures 4 and 5 illustrate extensive examples of interleaving TALE-SPIN stories with GPT-3. The following subsections discuss observations of the interleaved stories.

#### **4.1 Processing LLM Responses**

One common occurrence was that, although our intention was to get a response from the LLM consisting of a single sentence, frequently, the LLM response was often longer. Our system post-processes the LLM responses to obtain only the first sentence of the LLM response to be concatenated to the interleaved story. While the OpenAI API does allow the user to specify the maximum number of tokens in the response, we did not attempt to use this setting in completion requests to the model to control the response so that we would only receive a single sentence. Generally there was a wide range of response lengths from the API, from several sentences to no response at all (see the following section).

#### **4.2 LLM Nonresponse**

We frequently encountered cases where there was no response from the LLM to our completion requests (These are indicated “no response” or “None” in Figures 4 and 5). In these cases calls to the OpenAI API did return successfully, but with empty response content. We could not be certain about the reason for these non responses, and whether they might be caused by a communication error with an OpenAI server, a server error, or simply that the LLM itself chose not to respond.

We ran a number of brief experiments with both TALE-SPIN and human-generated prompts, and we found that the frequency of non-responses varied widely depending on the length of the prompt, and the “top\_p” and “temperature” parameters of the GPT-3 API, each of which controls randomness and creativity of the generated text. For some combinations, the frequency of non-responses was as high as 39%. We were able to significantly reduce the frequency of non-responses, mainly by reducing the lengths of the example stories in the prompts. We suspected that non-response was correlated with higher narrative complexity of certain stories, but could not develop sufficient evidence to confirm our suspicion. We note that the more complex narrative of our two examples (Figure 5) has two non-responses.

#### **4.3 Matching Styles**

We found that when interleaving, the LLM was adept at making contributions which matched the simple linguistic style of the example TALE-SPIN stories provided in the prompts (Figure 4). However, while matching the overall style of the TALE-SPIN stories, GPT-3 introduced words which

Original TALE-SPIN Story	TALE-SPIN/GPT-3 Interleaved Story
Joe was hungry.	TALE-SPIN: Joe was hungry.
Joe wanted not to be hungry.	GPT-3: Joe knew that Joe was hungry.
Joe wanted to have the honey.	TALE-SPIN: Joe wanted not to be hungry.
Joe wanted to know where the honey was.	GPT-3: Joe tried to find some food.
Joe thought that Joe did not know where the honey was.	TALE-SPIN: Joe wanted to have the honey.
Joe thought that Joe did not have the honey.	GPT-3: Joe saw the honey.
Joe wanted to have the berries.	TALE-SPIN: Joe wanted to know where the honey was.
Joe wanted to know where the berries were.	GPT-3: Joe asked Irving where the honey was.
Joe thought that Joe did not know where the berries were.	TALE-SPIN: Joe thought that Joe did not know where the honey was.
Joe thought that Joe did not have the berries.	GPT-3: Irving said that the honey was near the elm-tree.
Joe wanted to have the fish.	TALE-SPIN: Joe thought that Joe did not have the honey.
Joe wanted to know where the fish was.	GPT-3: Joe got the honey.
Joe thought that Joe did not know where the fish was.	TALE-SPIN: Joe wanted to have the berries.
Joe thought that Joe did not have the fish.	GPT-3: Joe saw the berries.
Joe knew that Joe was hungry.	TALE-SPIN: Joe wanted to know where the berries were.
	GPT-3: Joe asked Irving where the berries were.
	TALE-SPIN: Joe thought that Joe did not know where the berries were.
	GPT-3: Irving said that the berries were near the river.
	TALE-SPIN: Joe thought that Joe did not have the berries.
	GPT-3: Joe got the berries.
	TALE-SPIN: Joe wanted to have the fish.
	GPT-3: Joe saw the fish.
	TALE-SPIN: Joe wanted to know where the fish was.
	GPT-3: Joe asked Irving where the fish was.
	TALE-SPIN: Joe thought that Joe did not know where the fish was.
	GPT-3: Irving said that the fish was near the river.
	TALE-SPIN: Joe thought that Joe did not have the fish.
	GPT-3: Joe got the fish.
	TALE-SPIN: Joe knew that Joe was hungry.
	GPT-3: Joe had the fish.

Figure 4. An example of TALE-SPIN and GPT-3 interleaving.

were not present in the prompt stories. For example in one case GPT-3 responds “Irving said that the honey was near the elm-tree” (Figure 4) although the lexicon and simple generator for TALE-SPIN is only capable of generating using “told” as in “Irving told Joe that the honey was near the elm-tree.” The GPT-3 responses “Joe saw the honey”, “Joe got the honey,” are further examples of this; “saw” and “got” are not part of the Micro TALE-SPIN generation lexicon and were introduced

by GPT-3. In this way, the LLM was successful at embellishing the TALE-SPIN stories' language forms.

#### 4.4 Enforcing a Story Plot

One aspect of story coherence concerns whether the story has a plot and whether story events move toward a conclusion. One can see in Figures 4 and 5 the ways in which an LLM can “go in circles”, stating the same events ad infinitum.

In the following excerpt, GPT-3 repeats characters' efforts previously made by the TALE-SPIN story to divulge information in exchange for a desirable item. Namely, GPT-3 contributes a second instance in which Joe tries to attain information about the location of honey, as well as a second instance where Irving decides to give Joe information in return for the worm. It is interesting how, following TALE-SPIN's prompted “Joe gave Irving the worm.”, GPT-3 generates a near exact replication of the TALE-SPIN prompt that proceeded GPT-3's own earlier contribution of the same sentence, “Joe gave Irving the worm.”

TALE-SPIN: Irving told Joe that if Joe would give  
Irving the worm then Irving would tell  
Joe where the honey was.

GPT-3: Joe gave Irving the worm.

TALE-SPIN: Irving decided that if Joe would give  
Irving the worm then Irving would tell  
Joe where the honey was.

GPT-3: Irving told Joe that the honey was near  
the oak-tree.

...

GPT-3: Joe asked Irving whether Irving would  
tell Joe where the honey was if Joe gave  
Irving the fish.

TALE-SPIN: Joe gave Irving the worm.

GPT-3: Irving decided that if Joe would give  
Irving the worm then Irving would tell  
Joe where the worm was.

Similar circular behavior is known to exist with previous generations of deep learning models which perform image captioning (Dai et al., 2019). The interleaving of GPT-3 outputs with a fixed story from a symbolic story generator like TALE-SPIN forces the progression of the story.

#### 4.5 Commonsense

Consider the last line in the excerpt from the previous section:

Original TALE-SPIN Story	TALE-SPIN/GPT-3 Interleaved Story
<p>One day, Joe was hungry.  Joe wanted to know where the honey was.  Joe wanted to be near Irving.  Joe was near the oak-tree.  Joe asked Irving whether Irving would tell Joe where the honey was if Joe gave Irving the worm.  Irving told Joe that if Joe would give Irving the worm then Irving would tell Joe where the honey was.  Irving decided that if Joe would give Irving the worm then Irving would tell Joe where the honey was.  Joe wanted to be near the worm.  Joe was near the ground.  Joe had the worm.  Joe drove to the oak-tree.  Joe gave Irving the worm.  The worm was near Irving.  Irving told Joe that the honey was near the elm-tree.  Joe drove to the elm-tree.  Joe took the honey.</p>	<p>TALE-SPIN: One day, Joe was hungry.  GPT-3: Joe knew that Joe was hungry.  TALE-SPIN: Joe wanted to know where the honey was.  GPT-3: Joe searched for the honey.  TALE-SPIN: Joe wanted to be near Irving.  GPT-3: Irving searched for the honey.  TALE-SPIN: Joe was near the oak-tree.  GPT-3: Irving was near the elm-tree.  TALE-SPIN: Joe asked Irving whether Irving would tell Joe where the honey was if Joe gave Irving the worm.  GPT-3: Irving said yes.  TALE-SPIN: Irving told Joe that if Joe would give Irving the worm then Irving would tell Joe where the honey was.  GPT-3: Joe gave Irving the worm.  TALE-SPIN: Irving decided that if Joe would give Irving the worm then Irving would tell Joe where the honey was.  GPT-3: Irving told Joe that the honey was near the oak-tree.  TALE-SPIN: Joe wanted to be near the worm.  GPT-3: Irving decided that if Joe would give Irving the worm then Irving would tell Joe where the worm was.  TALE-SPIN: Joe was near the ground.  GPT-3: Irving was near the cave.  TALE-SPIN: Joe had the worm.  GPT-3: None  TALE-SPIN: Joe drove to the oak-tree.  GPT-3: Joe asked Irving whether Irving would tell Joe where the honey was if Joe gave Irving the fish.  TALE-SPIN: Joe gave Irving the worm.  GPT-3: Irving decided that if Joe would give Irving the worm then Irving would tell Joe where the worm was.  TALE-SPIN: The worm was near Irving.  GPT-3: Irving told Joe where the worm was.  TALE-SPIN: Irving told Joe that the honey was near the elm-tree.  GPT-3: None  ...  TALE-SPIN: Joe took the honey.  GPT-3: Irving decided that if Joe would give Irving the fish then Irving would tell Joe where the fish was.</p>

Figure 5. A more extensive example of TALE-SPIN and GPT-3 interleaving.

GPT-3: Irving decided that if Joe would give  
Irving the worm then Irving would tell  
Joe where the worm was.

Irving decides that in return for receiving the worm from Joe, he will tell Joe the location of that very worm. Even disregarding coherence to surrounding story structures, GPT-3's commonsense fails within a structure that it generates itself. Of particular interest is that such an instance where GPT-3 demonstrates internally faulty commonsense is one in which it is matching style to prior TALE-SPIN structures.

#### 4.6 Coherence

We were also interested in how an LLM might be able to embellish a story while maintaining logical story coherence. In many cases, GPT-3 provided content which made sense in between sentences of the TALE-SPIN story:

TALE-SPIN: Joe was hungry.  
GPT-3: Joe knew that Joe was hungry.  
TALE-SPIN: Joe wanted not to be hungry.

In this case GPT-3's response was both sensible and matched the style of a TALE-SPIN story.

In other cases GPT-3 attempted to move the story along. Below GPT-3's response described an action by Joe that was performed in service of a goal described in the previous segment from TALE-SPIN:

TALE-SPIN Joe wanted to know where the honey  
was.  
GPT-3: Joe asked Irving where the honey was.  
TALE-SPIN: Joe thought that Joe did not know where  
the honey was.  
GPT-3: Irving said that the honey was near the  
elm-tree.  
TALE-SPIN: Joe thought that Joe did not have the  
honey.

This response from GPT-3 also serves the function of introducing Irving as a character into the story. We note that Irving is not a character in the original TALE-SPIN story used for interleaving, but Irving is a character in other TALE-SPIN stories. Importantly, Irving is a character in the complete example story provided to GPT-3 in the prompt before the interleaving part of the prompt begins.

However, in many cases, GPT-3’s attempts to move the story along in its interleaved contributions conflicted with subsequent content in the TALE-SPIN story. For example, just before the previous segment is this pair of story events:

TALE-SPIN: Joe wanted to have the honey.  
GPT-3: Joe saw the honey.  
TALE-SPIN: Joe wanted to know where the honey  
was.  
...

GPT-3’s contribution, “Joe saw the honey,” clearly conflicts with the following several story events, since seeing the honey implies knowing where it is, and obviates the need for Joe to ask Irving where it is. This kind of conflict is not unexpected given that the TALE-SPIN story is static and does not adapt dynamically to the GPT-3 responses.

## 5. Conclusion

In this paper we presented an initial study of how a symbolic story generation system can be used to maintain coherence of a story when using a large language model for story generation. The interleaving method performs sentence-by-sentence combinations of the outputs of TALE-SPIN, one of the first well-known AI story generation systems, and GPT-3, a state-of-the-art LLM.

One major drawback of our the approach taken in these initial studies is that, while GPT-3 is a large language model which is able to adapt to interleaving of its own content with content from TALE-SPIN, the TALE-SPIN story content remains fixed. To create a link for TALE-SPIN to potentially adapt to GPT-3’s responses, it may be possible in future work to process GPT-3’s responses in some way to produce symbolic assertions which can be fed back into TALE-SPIN, making the TALE-SPIN contributions to the story dynamic.

Our larger goal is to explore how story generation systems may be a tool for mitigating unconscious cognitive biases that are, a critical barrier facing the diffusion of pro-social and pro-environmental innovations in society. Future work will compare interleaved stories with stories generated with an LLM alone. We will conduct studies with human participants who consume the stories to evaluate them for their logical narrative coherence, believability, and whether the stories are surprising or inspiring.

## 6. Acknowledgements

We thank the Toyota Research Institute for supporting this research.

## References

- Alabdulkarim, A., Li, S., & Peng, X. (2021). Automatic story generation: Challenges and attempts. *Proceedings of the Third Workshop on Narrative Understanding* (pp. 72–83). Virtual: Association for Computational Linguistics.
- Alhussain, A. I., & Azmi, A. M. (2021). Automatic story generation: A survey of approaches. *ACM Comput. Surv.*, 54. From <https://doi.org/10.1145/3453156>.
- Dai, M., Grandic, S., & Macbeth, J. C. (2019). Linguistic variation and anomalies in comparisons of human and machine-generated image captions. *Advances in Cognitive Systems*, 8, 33–51.
- Dale, R. (2021). Gpt-3: What’s it good for? *Natural Language Engineering*, 27, 113–118.
- Fan, A., Lewis, M., & Dauphin, Y. (2018). Hierarchical neural story generation. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 889–898). Melbourne, Australia: Association for Computational Linguistics.
- Floridi, L., & Chiriatti, M. (2020). Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30, 681–694.
- Meehan, J. (1981a). Micro TALE-SPIN. In R. C. Schank & C. K. Riesbeck (Eds.), *Inside computer understanding: Five programs plus miniatures*, 227–258. Hillsdale, NJ: Lawrence Erlbaum Associates.
- Meehan, J. (1981b). TALE-SPIN. In R. C. Schank & C. K. Riesbeck (Eds.), *Inside computer understanding: Five programs plus miniatures*, 197–226. Hillsdale, NJ: Lawrence Erlbaum Associates.
- Meehan, J. R. (1977). Tale-spin, an interactive program that writes stories. *Proceedings of the Fifth International Joint Conference on Artificial Intelligence* (pp. 91–98). Cambridge, MA.
- Schank, R. C. (1972). Conceptual dependency: A theory of natural language understanding. *Cognitive Psychology*, 3, 552–631.
- Schank, R. C. (1975). *Conceptual information processing*. New York, NY: Elsevier.
- Srivastava, A., et al. (2022). Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. From <https://arxiv.org/abs/2206.04615>.