# Automatic Story Generation: A Survey of Approaches

ARWA I. ALHUSSAIN and AQIL M. AZMI, King Saud University

Computational generation of stories is a subfield of computational creativity where artificial intelligence and psychology intersect to teach computers how to mimic humans' creativity. It helps generate many stories with minimum effort and customize the stories for the users' education and entertainment needs. Although the automatic generation of stories started to receive attention many decades ago, advances in this field to date are less than expected and suffer from many limitations. This survey presents an extensive study of research in the area of non-interactive textual story generation, as well as covering resources, corpora, and evaluation methods that have been used in those studies. It also shed light on factors of story interestingness.

CCS Concepts: • **Information systems** → *Information extraction*; • **Computing methodologies** → **Natural language processing**; **Discourse, dialogue and pragmatics**; **Natural language generation**; *Lexical semantics*;

Additional Key Words and Phrases: Text generation, story generation, datasets, evaluation, survey

**ACM Reference format:**
Arwa I. AlHussain and Aqil M. Azmi. 2021. Automatic Story Generation: A Survey of Approaches. *ACM Comput. Surv.* 54, 5, Article 103 (May 2021), 38 pages.
https://doi.org/10.1145/3453156



## 1 INTRODUCTION

Stories are a significant part of every culture, attracting people regardless of age. For that reason, stories have always been a medium for entertainment, moral lessons, and wisdom inspiration. In recent decades, stories have also been used as tools for assessing and educating children.

We may define creativity as the ability to generate novel and valuable ideas, where valuable means beautiful, interesting, and useful [26]. Generating stories using computers is a complex task of computational creativity, which lies in the area where psychology and **artificial intelligence (AI)** intersect. To teach computers how to generate a story, we need to understand how humans create one. Knowing this enables computer scientists to mimic the human brain. However, generating stories using computers helps psychologists better understand human cognition.

Many applications can benefit from automatic story generation, e.g., entertainment, where many stories can be produced with minimum effort [12, 107]. It can also be used for education where stories are customized to the learners' needs [13, 99]. In gaming, interactive stories play a major role in increasing the interestingness of games [67, 173].

This research was funded by the Deputyship for Research and Innovation, "Ministry of Education," in Saudi Arabia through project no. IFKSURP-91.
Authors' address: A. I. AlHussain and A. M. Azmi (corresponding author), Department of Computer Science, College of Computer & Information Sciences, King Saud University, Riyadh, 11543, Saudi Arabia; emails: {ahussain, aqil}@ksu.edu.sa.

0360-0300/2021/05-ART103 $15.00
https://doi.org/10.1145/3453156

Automated story generation is the problem of mechanically selecting a sequence of events or actions that meet a set of criteria and can be told as a story [90]. Each story has a story world, interacting characters, and objects. Furthermore, a story may have an author goal, which is the message that the author aims to deliver to the story receiver through the story events. Although story generation systems started in early 1960s [150], they did not achieve outstanding results and are still classified as weak AI systems because their creativity is not comparable to humans [75].

For a computer system to be creative, it must generate stories different from past seen ones. Many attributes must be considered, including story settings such as time and space, story characters, their desires, and plans to achieve these desires. In addition, the interactions between characters and conflicts that may occur between characters' desires are essential. These attributes result in an enormous growth of the story space, and hence searching for a story in this vast space is difficult, inefficient, or impossible. The large number of attributes makes story generation difficult, and accounting for its aim, believability, and interestingness further complicates the generation process. Open story generation, where stories are generated without relying on a priori engineered domain models, adds two extra challenges to story generation: the automatic construction of the domain model and evaluating the story progress to guide the generation process.

As a long-standing problem, computational narratives received many efforts to survey and classify story generation systems. In 2009, Gervás [60] reviewed how story generation systems emulated human creativity and to what extent they implement the key features of computational creativity. Since this work was published, computational narrative attracted the research community, and a large number of automatic story generators were proposed. Young et al. [187] conducted a survey centered on planning and reasoning in computational narratives, and a more recent survey by Kybartas and Bidarra [84] classified story generation systems based on degrees of automation of plot and space generation to four main categories: manual authoring, plot generation, space generation, and finally story generation that automates elements of both plot and space. In this survey, we aim to provide the reader with a comprehensive guide on automatic story generation. We survey computational approaches of story generation from an AI perspective and their intersection with cognitive science. We overview available knowledge sources for building story generation systems and the different evaluation metrics used in the literature. We also discuss the factors of story interestingness. No doubt there exist earlier surveys, e.g. [84, 187], some from not so long ago. However, we believe that our work can be seen as an up-to-date survey since it covers many newly proposed studies that were not covered previously.

Bailey [17] classified story generation systems into three main categories: *author models* that simulate the author's thoughts in generating a story, *story models* that generate stories by manipulating their structural artifact, and *world models* that generate stories based on the goals and plans of characters [106]. This article categorizes story generators into three main categories: structural models, planning-based models, and **machine learning (ML)** models. Unlike Bailey, we categorize author and world models as goal-directed approaches, which is a subcategory of planning-based models. However, our structural models are analogous to Bailey's story models, where story generation proceeds from an abstract representation of the story as a structural (or linguistic) artifact [17]. They often do not concentrate on causal relations between events. These models can be viewed as top-down approaches where story grammars are used to guide the generation process.

This report is organized as follows. Section 2 introduces some basic definitions of what constitutes a story. The three models used to generate stories are covered in Sections 3, 4, and 5. Section 6 overviews the knowledge needed by story generators. Factors of story interestingness are discussed in Section 7. Section 8 provides an overview of different evaluation methods.

A general discussion is presented in Section 9, and the conclusion and future directions are presented in Section 10.

## 2 DEFINITIONS

A *story* is a description of real or imaginary actors and events generated to achieve one or more goals, such as entertainment or education. Usually, stories have one or more *themes*, which are the central ideas of the story that the author conveys to the receiver [77].

A story *event* happens at a particular time and place [5] and transforms the world from one state into another. Each story has a *plot* representing the sequence of events and the causes affecting these events. Story *characters* are the actors of the story or those affected by them. Most short stories have only one primary character called the *protagonist* who is connected to most of the plot and events and the cause-effect relationships between these events. A story *space* includes the characters, settings, props, and anything present either physically or abstractly in the space of the narrative [84]. The *fabula* refers to a story world in which story events occur in chronological order. The *syuzhet* characterizes selected contents of the fabula arranged in a particular presentation order, taking into account the reader [15]. The *discourse* is the structure of how a story is organized into a surface expression [3]. It includes, but is not limited to, the syuzhet.

A *plot graph* is a representation used in story generation systems to model the plot space. After abstracting a story into discrete plot points, each of which represents some event in the story, some ordering constraints are assigned to plot points by placing them in a directed acyclic graph to define the space of possible sequences of events [180]. Additional disjunctive constraints can also be applied to specify the story points that never occur together [90, 117]. A *script* is a structure that describes an appropriate sequence of events in a particular context [153]. It represents a story as a sequence of slots and imposes constraints of what can fill these slots. The slot content can also be affected by the content of the other slots of the script.

The story *frames* are structures used to represent different story elements. For example, a character frame stores specific information about a character, such as a name, role, and status [86]. However, *events* frames formalize the attributes and constraints of an event, such as actors, locations, and a list of possible actions [31].

For a more realistic appearance, some supplementary *settings* should be added, such as story time and place. Moreover, to achieve the story goals and produce an interesting story, the story plot should be well structured. Many story structure models exist for analyzing and generating stories. However, the most well known and widely accepted story structure is Freytag's pyramid [57], which can be broken down into five main components: *exposition*, where the main characters and story settings are introduced; *rising action*, where the events start to happen that leads to the story climax; the story *climax*, which is the point where the main action or highest tension of the story occurs; and *falling action*, which is the sequence of events leading from the climax to the story *resolution* where the story's main problem is resolved and the story ends. As we can see from Freytag's pyramid, story structure is what distinguishes a story from other types of literature. Its importance lies not only in achieving story goals but also in increasing the interestingness of the story.

The believability of a story is highly affected by its *consistency*, where the sequence of events seems logical to the receiver. Three factors contribute to story consistency: the cause-effect relationship between events, the conformity between the events and the story world, and the accordance between the characters' personalities and actions. Once the story loses its consistency, it will lose its attraction. Therefore, story consistency can be considered as necessary as story structure and is an essential requirement for any generated story [169].

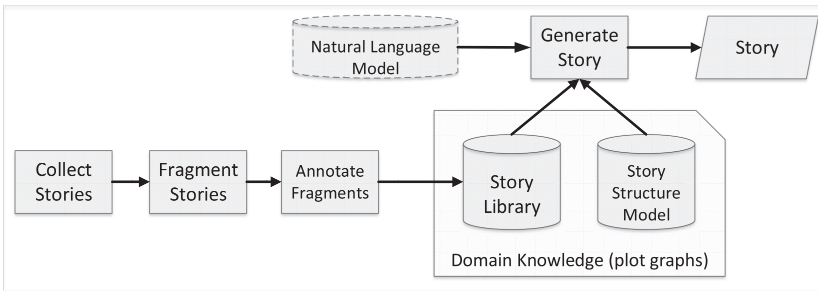


Fig. 1. A framework of structural story generation.

## 3 STRUCTURAL MODELS

In cognitive science, story grammar theories view stories as scripts, where a script is a structure that describes an appropriate sequence of events in a particular context [153]. These theories arose from the fact that in the real world, events usually occur in stereotypical patterns. For example, when someone wants to eat at a restaurant, the sequence of events is as follows: enter restaurant, sit at table, read menu, order food, eat food, pay money, and leave restaurant. These sequences, or patterns, are used as schemas for guiding story generation. However, a context may have different scenarios that vary slightly, resulting in different interfering patterns. For instance, in the restaurant context, the sequence of events may differ if there is no menu on the table. Another scenario occurs when a customer enters a fast-food restaurant where payment precedes eating. The diversity of patterns enhances the diversity of the generated stories.

In the field of story generation, schemas are employed to automatically generate structured stories by dividing the stories into slots following a given schema. The slots are then filled by pasting similar slots of previously collected and annotated stories, considering the inter-effect between the generated story slots' contents. As Figure 1 shows, structural story generation starts by annotating story fragments, and then the annotations are used to "glue" the fragments together based on plot graphs or story grammars.

One of the earliest and most widely adopted efforts to formalize stories into structural models is the Russian formalist Vladimir Propp's work. In his book *Morphology of the Folktale*, Propp [135] analyzed nearly 600 Russian folktales. He concluded that all folktales are composed of the same 31 character actions, which he called *functions* such as absentation, villainy, lack, struggle, and victory. These functions may not appear in every tale; however, functions appearing in one tale follow a rigid order. Propp's structure was widely used for automatic story generation, especially in early research.

Joseph Grimes' forgotten pioneer system recently came to light [150]. Grimes implemented Propp's story morphology by randomly selecting a subset of Propp's 31 functions and then ordering them based on Propp's grammar. In addition, Grimes' system used intelligent referring expressions. For example, "a lion" that was mentioned at the beginning of the story is referred to as "the lion" in the remainder of the story. It also used the discourse marker "thus" to connect the final sentence to prior sentences of the story. Table 1 shows a sample story generated by Grimes' system.

In addition to Propp's functions, there were many efforts to formalize stories. In his book *The Thirty-Six Dramatic Situations*, Polti [134] created a descriptive list to categorize every dramatic situation that might occur in a story or performance [154] by analyzing Greek texts, French, and some non-French works. Although Polti did not propose a way to combine the dramatic situations

Table 1. The Only Surviving Output of J. E. Grimes 1960s' Computer-Generated Story as Reported by Ryan [150]

| |
|---|
| *A lion has been in trouble for a long time. A dog steals something that belongs to the lion. The hero, lion, kills the villain, dog, without a fight. The hero, lion, thus is able to get his possession back.* |

to generate stories, his work was adopted by some narrative generation systems as in the work of Jhala and Young [78].

### 3.1 Graph-Based Approaches

The simplest form of a script used in story generation is to build story graphs. During the design phase, a branching story graph representing all possible stories' space is constructed. Then, at the generation phase, the graph is traversed to find a linear path that represents the generated story. The quality of the generated story depends mainly on the quality of the constructed graph. In addition, adding some constraints on the graph search, such as path length range, can enhance the results. Based on Propp's story structure, Maranda [104] developed a graph that can generate folktales. Maranda's graph consists of nodes that contain Propp's functions with specific start nodes and termination nodes. The Russian folktales annotated knowledge base is searched for a piece that matches the node function at each traversed node. The retrieved piece is then concatenated to the generated story. The process is repeated until we reach a terminal node. This approach not only generates the original Russian folktales but can also generate new stories. However, Maranda's graph is cyclic, which may lead the system to infinite iterations.

The SCHEHERAZADE system proposed by Li et al. [90] collects human experiences about a topic domain in the form of scripts. Then, it learns a plot graph based on these scripts. The graph is then traversed to generate stories. SCHEHERAZADE graphs are similar to Maranda's graph in both having specific start and termination nodes. However, its graphs are acyclic and apply mutually inclusive constraints on some of the generated stories' events.

### 3.2 Grammar-Based Approaches

Using grammar to generate stories started when Lakoff [85] reformulated Propp's story structure into a story grammar. Viewing stories as words of the narrative's formal language where the alphabets are Propp's functions, Lakoff used expandable rewrite rules to generate stories, which meant different stories could be produced by selecting different expansions. His work inspired researchers into proposing other story grammars. As a further example, Pemberton [127] proposed a story grammar for an old French epic, which was implemented later as GESTER [128], a program that generates story models based on Pemberton's grammar. Its stories have a clear beginning, middle, and end. BRUTUS [31] is also a system that generates betrayal stories based on story grammars. It created complex stories based on frame structures, where every element of the story, such as characters and events, is considered story frames. These frames are then grouped into several story themes. All previously mentioned story grammars are specialized grammars limited to the domain that produced them. Thus, they can only generate a small restricted set of stories, which necessitates a call for more general story grammars.

Theoretically, Rumelhart [149] was the first to propose general story grammar. Since then, several general story grammars were proposed, including that of Thorndyke [170], which was widely adopted due to its simplicity. To conclude, story structural models are easy-to-implement, fast-to-implement approaches that can generate well-structured stories provided that the structural model is well structured. They can also generate interesting stories. Nevertheless, as discussed previously, structural models focus on the structure of the story, i.e., they focus on the syntax of the story rather than the semantics, whereas stories are semantic models in nature. Therefore, the logical relationships between story events and between character intentions and actions will be negatively affected, affecting story coherence and believability. In addition, structural models are rigid; they can only generate stories that satisfy the provided story structure and cannot modify their knowledge to generate different stories. Their generated stories are limited to one protagonist because having more than one protagonist requires complex logical relationships. Finally, story structural generation suffers from the over-generation problem. In other words, it generates non-story texts and accepts them as stories. This includes procedural exposition and stories that are ill formed [23].

## 4 PLANNING-BASED MODELS

Story grammar theories were criticized as a story understanding approach [23, 24]. According to psychologists, story grammars can build a story syntactically. However, they do not account for the semantic relationships in the story. Therefore, they cannot be applied to stories with conflicting goals or with multiple protagonists. The worst-case scenario of story grammar is when they accept non-stories as stories [23]. The story points theory was proposed as a response to these criticisms. Here, we view a story as a chain of causally connected events to pursue an end goal [181]. Having the causal connection between events makes more sense to the reader and serves the story semantics. To aid story writers, Cook [46] published *Plotto: The Master Book of All Plots*, which contains 1,852 numbered plot fragments, each of which refers to several potential predecessors and successors with formal instructions on how to combine them to produce complete plots. It also introduced three different ways to start with a story. The wide variety of plot fragments and the structured way of connecting the fragments made Plotto an interesting source for computational narratives. Cook's approach follows the story points theory, where the focus is on the logical flow between successive fragments rather than on the story's overall structure.

Eger et al. [53] proposed Plotter, a computational story generator that generates plots from Plotto's fragments and their associated instructions. However, many generated stories were inconsistent and had logical conflicts. This limitation is considered inherent to the story points approach, where each step only depends on the current state of the story and not past states. To improve story consistency, the authors suggest using an AI planning operator representation of each fragment. Indeed, story points theory has been popular for automatic story generation using AI planning algorithms. Knowing that both theoretical story planning and AI planning are based on reasoning, the analogy between them seems obvious. In general, generating stories using AI planning works by providing an initial state and a goal for a reasoner that infer actions and ultimately lead the initial state to the story goal. An optional directing process may be introduced to enhance the quality of the generated story, as shown in Figure 2. The following sections review the different approaches for employing AI planning in automatic story generation.

### 4.1 Goal-Directed Approaches

Goal-directed approaches were the first intelligent story generators, followed by structural models. Using the goal-based agents, which range from simple atomic problem-solving agents to structured planning agents, story planners were used in a wide range of story generators in the literature.

*4.1.1 Simulation Approach.* Meehan [109] was the first to introduce AI for automatic story generation in his pioneering project TALE-SPIN. In contrast to story grammars, TALE-SPIN concentrated on characters' needs and intentions to fulfill these needs using AI problem-solving

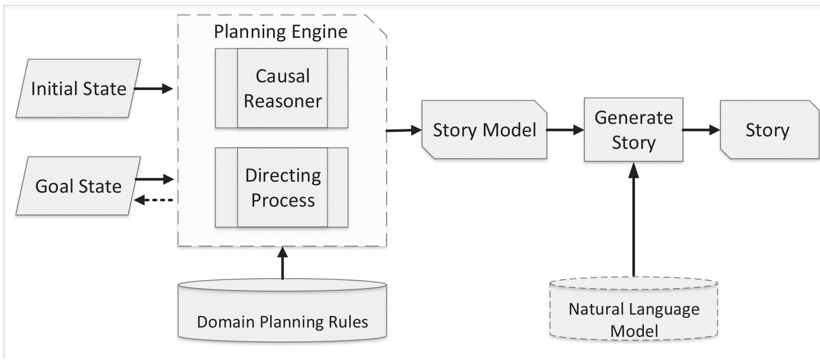


Fig. 2. The framework of goal-directed story generation.

techniques. Through building a world simulation planner, TALE-SPIN was entirely driven by character goals. Story generation starts by setting an initial state, i.e., a description of the story world and one or more character goals. Then, by using an inference engine (reasoner) to implement a forward chaining algorithm, the story plan is produced by inferring chains of causal events considering the effect of each event on the story world. The process continues until a character goal is reached. This approach gives story characters clear intentions and thus improves their believability. TALE-SPIN was able to generate short coherent stories similar to Aesop's fables. For a well-structured story, the world's initial state and the character goals should be declared. However, focusing on a character's own needs and actions to fulfill these needs may result in uninteresting stories that lack climax or resolution. Another shortcoming of TALE-SPIN is that many of its good generated stories reassemble the source stories used to build its rigid knowledge base.

*4.1.2 Global-Schema Approach.* In story writing, authors aim to write coherent and exciting stories by creating a set of goals that form the story's skeleton; then, they direct story characters to pursue these goals. To improve the structure of the story, its generation systems have simulated this process by moving the focus from character goals to author (global) goals, which are independent of the story characters. Because authors are not part of the story world, they are never threatened or involved in any competition, and they do not benefit from opportunities. Therefore, the authors' intention as independent agents is to produce a good story without giving an advantage to a specific character. Moreover, author goals may result in a goal competition or goal conflict between the different characters in the story, which increases a story's interestingness.

Dehn [47] proposed AUTHOR, one of the first story generation systems based on author goals. It implements a reconstructive dynamic memory architecture that simulates a human author by using an external paper to write a story draft and revise it several times before it can be finally delivered. The story generation process starts with a set of author goals that are provided as input. Then, a loop of three subtasks begins. First, dredging starts by searching the memory for related materials. Then, milking occurs by selecting the most appropriate part of the retrieved materials. Finally, conceptual reformulation takes place by revising the author's goals and modifying them if needed. The loop continues until all of the author's goals are satisfied. The stories generated by AUTHOR are generally well structured and more interesting than those generated by a character-goal-based system. Nevertheless, the characters' believability was negatively affected because they (the characters) sometimes acted without clear intentions to satisfy the author's goals.

*4.1.3 Multi-Agent Approach.* In the interest of generating well-structured coherent stories, researchers have aimed to direct story generation by both character goals and author goals. This approach is similar to the character-goals approach because characters are directed by their own goals, which preserve story coherence. Riedl et al. [139] created Automated Story Director to guide characters' actions throughout the story generation process to avoid the generation of a poorly structured story. The Virtual Storyteller [169] is a computational narrative that uses intelligent agents to generate stories. Although characters plan to achieve their own goals, a virtual director agent who has general knowledge about plot structure is proposed to direct character actions to preserve the story's simple structure: a beginning, a middle, and a happy end. This is achieved by leading the plot in the desired direction using an environmental control, e.g., introducing new characters, and motivational control, e.g., introducing new character goals.

Riedl and Young [141] proposed FABULIST, an **Intent-driven Partial Order Causal-Link (IPOCL)** planner, which consists of two mechanisms. The first mechanism is the partial order causal-link planner that infers chains of characters' causal actions driven by the author's global goals. The second mechanism aims to preserve characters' believability by simulating the audience intention recognition process [34]. This mechanism is a unique reasoning process integrated into the planner that takes character actions and tries to predict the character intention (goal) based on these actions. If a character goal is not predictable, character actions will not be considered intentional, and thus the plan will be regarded as flawed and will need to be revised to ensure character believability. However, highly predictable goals will result in an uninteresting story. The IPOCL planner is slow, a big drawback. It takes approximately 12.3 hours to generate a complete plan [142]. In addition, the fact that IPOCL uses a non-standard representation language limits its improvements based on off-the-shelf planners that have faster performance.

To tackle intent-driven narrative planning by classical planners instead of the IPOCL specialized planner, Haslum [68] remodeled the narrative planning problem to embed character intentionality into it. He modeled character intentions as part of the narrative planning problem specification. Then, he used the intentions as preconditions of character actions. This compilation permits the use of off-the-shelf planners to generate stories and accelerate the generation process. Another extension of IPOCL is the **Conflict Partial Order Causal-Link (CPOCL)** planning algorithm proposed by Ware and Young [179]. CPOCL creates a model of conflict and then enforces the generation of conflict in stories by constraining the planner. This is done by using non-executed steps to model thwarted character intentions, which allow for partially executed plans.

### 4.2 Analogy-Based Approaches

The computational analogy is an AI approach based on the human cognitive process of analogy making. It operates by identifying similarities and transferring knowledge between a source domain and a target domain [191]. Using the analogy, a new problem can be solved by applying the solution to a previously known similar problem. This approach is applied in story generation systems by searching the knowledge base for a story world state similar to the current story world. Then, its next story event will be the next event for the generated story (Figure 3). The similarity measure differs between different systems.

MINSTREL [172] is one of the earliest analogy-based systems for story generation. It is a complex system driven by character and author goals where **Case-Base Reasoning (CBR)** is used mainly to achieve character goals. MINSTREL stores scenes (cases) in its episodic memory, where scenes are indexed by salient cues such as location and action to form groups of related scenes. When the story theme schemas instantiation fails because there are no matching scenes in the memory, MINSTREL creates novel scenes by using Transform-Recall-Adapt Methods (TRAMS). First, to simplify the search in the episodic memory for similar scenes, story schema specifications are transformed into a general form by substituting actors and objects with "someone" and "something," respectively. After recalling similar scenes from the episodic memory, retrieved scenes are

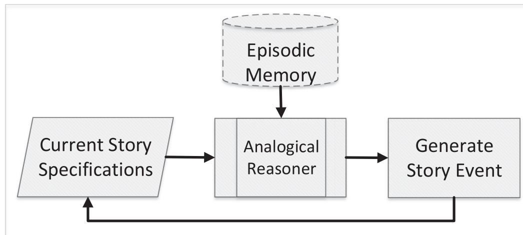


Fig. 3. The framework of analogy-based story generation.

Table 2. Sample Story Generated by MINSTREL [172], a Planning-Based Model for Story Generation

| |
|---|
| The Proud Knight 13 |
| *It was the Spring of 1089, and a knight named Godwin returned to Camelot from elsewhere. A hermit named Bebe told Godwin that Bebe believed that if Godwin jousted then something bad would happen. Godwin was very proud. Because Godwin was very proud, Godwin wanted to impress his king. Godwin jousted. Godwin lost the joust. Godwin hated himself.* |
| *Moral: Pride goes before a fall.* |

Table 3. Story Abstractions

| System | Abstraction Type | Story Abstraction |
|---|---|---|
| [36, 37, 76] | Pair | (`verb`, `dependency`) |
| [18, 19] | Triple | (`argument`$_1$, `relation`, `argument`$_2$) |
| [131] | 4-tuples | `verb`(`subject`, `object`, `prepositional`) |
| [133] | 5-tuples | (`verb`, `subject`, `object`, `prepositional`, `preposition`) |
| [105, 167] | 4-tuples | (`subject`, `verb`, `object`, (`propositional object` \| `indirect object` \| `causal complement` \| `unclassifiable dependency`)) |
| [6] | 5-tuples | (`subject`, `verb`, `preposition`, `object`, (`modifier` \| `prepositional object` \| `indirect object`)) |
| [186] | 1 word | The most important word of each sentence |

adapted to match the original story specification. It is worth pointing out that MINSTREL avoids using a scene more than twice to ensure the story's novelty. Table 2 shows a sample story generated by MINSTREL.

MEXICA [129] is a story generator based on the engagement-reflection cognitive account of writing [159]. For each character, MEXICA creates a story world context to record emotional links with other characters and the dramatic tensions produced in the story. These tacit elements work as pre- and post-conditions for story actions. In its long-term memory, MEXICA stores different schemas of tacit elements, and each schema is associated with a set of possible subsequent actions obtained by analyzing previous stories. During the engagement process, the long-term memory is searched for a schema that matches any of the story world contexts. If a matching schema is found, one of the actions associated with it will be selected as the next action of the story, and the story world contexts will be updated based on the selected action. During the reflection process, MEXICA reviews the generated story and evaluates its consistency, novelty, and interestingness compared to previous stories. If these requirements are not satisfied, guidelines will be produced to work as filters of actions when the engagement process starts again.

ProtoPropp by Gervás et al. [61] applies CBR to generate stories based on Propp's story morphology. To create the case base, stories were analyzed and annotated according to Propp's 31 functions, and an ontology of these functions was created to model the temporal relationships and co-occurrence constraints of the morphology. The story generation process is an interactive process that requires progressive user inputs. These inputs include the functions included in the story in addition to other attributes such as characters, roles, and functions. Swanson and Gordon [166] explained in detail how CBR can be applied for textual storytelling. This includes collecting and annotating the case library automatically, the similarity measures and ranking modules used in the retrieval process, the adaptation algorithm used to map the retrieved case to the story context being generated, and the evaluation of different system components.

### 4.3 Heuristic Search Approaches

To increase the variety of generated stories, researchers have expanded the stories search domain. However, as the search space grows, it becomes difficult for traditional planning techniques to find a solution efficiently. Therefore, heuristic search techniques were introduced. HEFTI [124] uses genetic algorithms to generate stories by portioning each story into timesteps that are represented by story components. Each story component is encoded into a chromosome where the genes are story elements, including agents, events, and objects. The fitness of a chromosome is calculated by summing the fitness attribute of each of its story elements that are manually evaluated by authors.

McIntyre and Lapata [108] applied genetic algorithms to generate stories. They first extracted plot graphs from the story corpus where each node represents a single event of the story associated with its arguments, such as nouns, adverbs, or adjectives. Then, an initial population of stories was created by sampling the story graph. To generate new chromosomes, a single point crossover is applied. The mutation is then applied either by replacing an event of the story, i.e., a node, or by replacing one of the event's arguments with a semantically similar argument. The chromosome's fitness is based on its coherence, which is calculated using the entity-grid document representation measure for local coherence [20]. Kartal et al. [80] formulated story generation as a Monte Carlo tree search problem. Each node in the tree represents a story state, and each edge represents an action that changes a state to a possible successive state. The best-generated story is chosen based on an evaluation function that considers two factors: the percentage of user goals accomplished by the story and the product of the believability of each action. The latter is a user-defined measure.

## 5 ML MODELS

Interest in the field of story generation never sustained a continuous interest. The past few years witnessed several research attempts to use ML for story generation. Looking at a story as a sequence of events, ML learns the conditional probability distribution between story events from a story corpus. The work in this field can be categorized as follows:

*Script learning and generation*: A system learns to predict missing script events based on other events of the script.
*Story completion*: A system learns to generate the missing event based on other story events.
*Story generation*: A system in which the system generates the complete story.

Most of these systems employ **Recurrent Neural Networks (RNN)**. RNN has been successful in **sequence-to-sequence (Seq2Seq)** problems such as machine translation [43, 165] and dialogue systems [171, 190]. For story generation, we can train RNN to predict a story event based on other story events. RNN can also be used to predict story sentences word by word based on language models. We will review the different approaches in applying ML for story generation.

### 5.1 Story Abstraction

Stories in their textual form contain many details that are insignificant for the story plot and add more dimensionality to the learning process. Therefore, it was essential to create a story abstraction that simplifies story representation and, at the same time, increases the potential overlap between stories [105]. This reduced the sparsity of stories and allowed for more efficient learning and inference. The most widely used story abstraction was to create a simple representation for story events that focus on the story's chain of events and its main entities. However, there is a trade-off between events representation simplicity and the extracted chain events coherence.

Chambers and Jurafsky [36, 37] and Jans et al. [76] represented stories as a chain of events, where each event is associated with the grammatical role played by the protagonist in `(verb,dependency)` pairs, i.e., subject-verb and verb-object pairs. They argue that verbs sharing co-referring arguments are semantically connected. For example, the subject of the verb "fall" has a high potential to occur as the subject of the verb "injured," which in turn has a high possibility to appear as the object of the verb "heal." This representation can be used to learn event schemas and to induce event schemas in a domain-independent manner. However, it may cause inconsistent subject-verb-object tuples because although the subject-verb and verb-object pairs are well formed, the entire tuple is not. It is also limited to one protagonist and cannot represent interactions between multiple entities. It cannot infer, for example, that when "X emails Y" there is a high potential that "Y emails X" too.

Instead of `(verb,dependency)` pairs, Balasubramanian et al. [18, 19] used an Open IE system to extract Rel-grams schemas, relational triples in the form $(\texttt{arg}_1, \texttt{Relation}, \texttt{arg}_2)$ where the relationship is a head verb and any prepositions are in addition to optional embedded nouns. The $\text{arg}_1$ and $\text{arg}_2$ are arguments represented as head nouns annotated with their semantic types. Although this representation increases sparsity compared to pair models, it achieved more coherent schemas.

Pichotta and Mooney [131] proposed a 4-tuple event representation in the form `verb (subject, object,prepositional)`. By including coreference relationships between different verb arguments, this model was able to express interactions between entities, which improves prediction accuracy. Later, Pichotta and Mooney [133] used a similar representation but considered prepositions. Compared to their earlier work [131], the system prediction accuracy was improved by implementing this representation. Inspired by that earlier work [131], Martin et al. [105] proposed a 4-tuple event representation in the form `(subject,verb,object,modifier)`, where the modifier can be a propositional object, indirect object, causal complement, or any other unclassifiable dependency. The same event representation is used by Tambwekar et al. [167]. A similar event representation is used by Ammanabrolu et al. [6], where each event is represented as a 5-tuple in the form `(subject,verb,preposition,object,M)`, where M can be a modifier, prepositional object, or indirect object. In the work of Yao et al. [186], a story abstraction was created by extracting each sentence's most important word.

### 5.2 Script Learning and Generation

Script learning and generation was the first step in generating stories using story corpora. This method aims to determine to what extent a given event and a set of events are related. Such a statistical model can help to predict new events belonging to a given chain of events. In general, statistical story learning systems proceed as follows:

(1) Run a dependency parser to extract verbs and their arguments.
(2) Run a coreference resolver to find all expressions that refer to the same entity in a story.
(3) Extract the sequence of events, AKA event chains, from a story based on the previous steps.
(4) Build a statistical model of the events chains.
(5) Use the statistical model to infer events of a story chain.

In their seminal work, Chambers and Jurafsky [36] used coreference relationships to extract the chain of events for a single protagonist. Then, they used **Pointwise Mutual Information (PMI)** to extract the pairwise relationships between events, as shown in Equation (1):

$$\mathrm{pmi}(e(w,d), e(v,g)) = \log\left(\frac{\Pr(e(w,d), e(v,g))}{\Pr(e(w,d)) \cdot \Pr(e(v,g))}\right), \tag{1}$$

where $e(w,d)$ is the verb-dependency pair (Section 5.1). After calculating the pairwise relationship scores, the most probable missing event of a chain of events can be found by selecting the event from the training corpus that maximizes the PMI if given all events in the story's chain of events. For temporal ordering of events, a support vector machine was trained to classify the temporal relationship between two events as in the work of Chambers et al. [38] but focusing on the "before" relationship. Given a chain of events, this system generates a ranked list of possible events that can fit as part of the chain.

Jans et al. [76] proposed skip $n$-grams for learning about events-chain statistics by pairing each event with the three following events in the chain. This approach outperformed the PMI method [36]. It is based on the observation that closely semantically related events do not necessarily appear next to each other. This approach also decreased data sparsity and hence improved the training process. Their bigram probabilities ranking function scores an event based on its position in the chain by considering the preceding and following events, which models an event chain in the order it was observed. Given an insertion point $p$ an event is scored as follows:

$$S(a) = \sum_{i=1}^{p-1} \log \Pr(a \mid a_i) + \sum_{i=p}^{|A|} \log \Pr(a_i \mid a), \tag{2}$$

where the conditional probability is given by

$$\Pr(e_1 \mid e_2) = \frac{C(e_1, e_2)}{C(e_2)}. \tag{3}$$

Balasubramanian et al. [18, 19] proposed the Rel-grams system, a Markov model system similar to that of Jans et al. [76] but focusing on relationship co-occurrence rather than argument co-occurrence. Given a relational triple in the form $(\texttt{argument}_1, \texttt{relationship}, \texttt{argument}_2)$, the Rel-grams system can predict one of the arguments if provided with the relationships and the other argument.

Pichotta and Mooney [131] proposed structured events with multi-arguments. By incorporating coreference information between the arguments of different events, they were able to encode the pairwise entity relationships between story events and therefore model the interaction between various entities. Their event structure enabled them to generate an event chain for the whole story instead of the separate entity-based event chains produced by verb-dependency pairs. However, the complexity of the events structure adds to the complexity of the statistical model. In verb-dependency chains, it is straightforward to calculate the co-occurrence of events by counting the number of times two events co-occur in the same chain. The co-occurrence of structured events in the work of Pichotta and Mooney [131] was calculated by counting the number of times two events occur in the same chain if, and only if, they have overlapping arguments. After counting the events co-occurrence, a scoring function similar to that of Jans et al. [76] is used. This system shows better prediction accuracy compared to systems with verb-dependency pairs.

Unlike previous count-based techniques, several studies have attempted to predict events using language models that involve compositional representations of events. Rudinger et al. [148] trained a Log-Bilinear model to predict story events. They argued that event prediction could be productively reframed as a language modeling task. Their discriminative language model showed improved performance compared to prior count-based methods. Pichotta and Mooney [133] used **Long Short-Term Memory (LSTM)** RNN to learn stories statistically. Their model was able to predict nouns or coreference information concerning event arguments. Looking at a story as a sequence of events abstracted as 5-tuples, they trained the LSTM model to anticipate the tuple's next element given the preceding element. The first element of the tuple is predicted based on the last element of the previous tuple. This model has shown better performance compared to several baseline systems. The same researchers extended their work to predict events directly from raw text without using explicit event structures [132]. Their experiments showed that the difference between raw text models and structured events models is marginal, and this indicates that extracting events structures is not necessary for event prediction, particularly in an encoder-decoder setup. Granroth-Wilding and Clark [63] compared several approaches for deriving vector representations of event predicates and argument nouns. The vector representations were fed into a compositional neural network model that predicts how probable it is that two events will appear in one event chain by performing a non-linear composition of their predicates and arguments.

Mostafazadeh et al. [114] proposed the **Story Cloze Test (SCT)** and built the ROCStories Corpora for testing the ability of machines to select a correct story ending given the story context. A detailed description of this work is presented in Section 8. Some researchers used ROCStories to build classification models that can choose the correct ending of stories based on various aspects. These can be categorized in general into feature-based models and neural models.

Chaturvedi et al. [39] proposed a script learning model based on three semantic aspects: event-sequence, emotional trajectory, and topical consistency. Although their work outperforms previous approaches, the researchers suggest using a thorough analysis of human behavior and societal norms to improve script learning. A similar feature-based model was proposed by Lin et al. [95]. Mostafazadeh et al. [116] trained a simple embedding model to predict the correct story ending based on the story context's embedding and the two alternative endings. Wang et al. [175] used generative adversarial networks, where the generative model generates a fake sample conditioned on the story context, and the discriminative model discriminates the real sample from the fake one. The discriminator has three models: an LSTM-RNN model to represent the sentence, an attention-based LSTM-RNN model to represent the document, and a bilinear model to calculate the context document and target sentence similarity. Notable enhancements on the SCT results were achieved when training on huge data as reported by more recent studies [41, 74, 91, 93].

### 5.3 Story Completion

Unlike previous research that predicts new story events by scoring known events, story completion aims to complete the plot when given a story context [65]. Most systems in this category conclude stories by generating a story ending based on previous story events.

Roemmele et al. [145] used the Children's Book Test (CBT) dataset as a story corpus. Story generation starts by taking an initial story that contains 20 sentences as input and generating the next sentence based on CBR. Then, RNN is used to generate the last sentence word-by-word comparing with the original 21st sentence as a gold standard. This study's main contribution is that it used several linguistic metrics to automate the evaluation of the generated stories. In addition, Hu et al.

[72] proposed a context-aware hierarchical LSTM model that can predict future subevents given previous subevents. This model generates a sequence of words describing the future subevent. It considers two levels of the event sequence: the sequence of words and the temporal sequence of events. It also considers the story topic as an additional contextual feature.

Li et al. [92] proposed a Seq2Seq model trained using adversarial training to generate diversified story endings. They argued that traditional Seq2Seq models, trained purely by maximum likelihood estimation, are suitable for generation tasks where a gold standard exists. Nevertheless, this is not the case in story ending generation, where every proper ending is acceptable. To improve the quality of generated endings, the generator is encouraged to create endings similar to story endings written by humans. Therefore, a discriminator (a binary classifier) is trained to label the output as human generated or machine generated. This classification is used as a reward for the generator in the reinforcement learning algorithm. Zhao et al. [189] improved the accuracy and fluency of generated story endings by applying the copy and coverage mechanism to the traditional Seq2Seq model proposed by See et al. [156]. To avoid the **out-of-vocabulary (OOV)** problem, the copy mechanism is used to generate story endings directly from previous story events via pointing. The coverage mechanism is used to overcome the repetitive words problem by maintaining a coverage vector that keeps track of the attention history to adjust future attention. A new objective function of semantic relevance loss was added to maximize the semantic relevance between the generated ending and the story. It is calculated as the cosine similarity between the plot semantic vector and the semantic vector of the generated ending. Although the semantic vector of the generated ending is the encoder's last hidden output, the plot semantic vector is calculated as proposed by Ma and Sun [102]. The generator was trained with a reinforcement learning algorithm that uses different evaluation metrics as reward functions to simulate the process of story generation by humans.

Guan et al. [65] proposed a neural model that generates a story ending considering two perspectives: story consistency and story implicit knowledge. All story events, attributes, and causal relationships between events play a role in story consistency. Therefore, story context clues were implemented by incremental encoding to maintain consistency. To mimic the human brain, which tends to understand a story and infer information based on its background knowledge, this model employs ConceptNet as a source of implicit knowledge and controls this knowledge through multi-source attention. This model was able to generate consistent story endings.

Wang et al. [176] proposed a model based on GPT-2 [137] to generate the missing parts of an incomplete story by conditioning the generated sentence on a previous sentence and a next sentence. Their model was able to create coherent stories that adhere to the provided end. Similarly, Wang and Wan [178] proposed a model for generating the missing story plot at any position for an incomplete story. Unlike the model of Wang et al. [176], this model can generate a sentence at the end of the story. It was adapted from the Transformer [174] by using shared attention layers for the encoder and decoder. BERT (Bidirectional Encoder Representations from Transformers) was used as the coherence discriminator. BERT is a new language representation model that is designed to pre-train deep bidirectional representations from unlabeled text [51].

### 5.4 Story Generation

Researchers were motivated to use Seq2Seq models to generate complete stories due to its success in different NLP tasks. Jain et al. [75] combined two off-the-shelf systems to construct a story generator that generates stories when given a sequence of independent short descriptions. First, Statistical Machine Translation (SMT) was used to translate phrases independently within a sentence. Next, a deep RNN was implemented to encode each sentence as a unit and then decode

Table 4. Sample Story Generated by a Data-Driven Model [75]

| |
|---|
| *On their trip to location, they arrive in front of a river. They decide to check out the city. They think its too packed with people, so they go sight seeing. The indoor poor temps them, but they decide not to jump in. They come across some ducks.* |

them into comprehensive stories. Although this system was able to generate story-like summaries, the summaries were not fully semantically related to the input description. The overall scores of the applied evaluation metrics were not very high. Table 4 shows a generated sample story.

Choi et al. [44] trained an RNN model to generate stories by predicting the next sentence. The model consists of two sub-models: RNN Encoder-Decoder (RNNED), which maps a sentence into a vector representation and vice versa, and RNN for Story Generator (RNNSG), which uses previously learned vectors to predict the next vector in the vectors sequence. The RNN model works by encoding story sentences into vectors and then using them to predict the next vector. The predicted vector is decoded into a sentence representing the next sentence of the story. The model was able to generate sentences with correct grammar and overall content. However, it misused some words in the generated sentences. Harrison et al. [66] used RNN to guide Markov Chain Monte Carlo (MCMC) sampling in generating stories, similar to the two-step process for generating stories in the work of Choi et al. [44]. It starts by reducing the natural language sentences into an event representation that contains a subject, verb, object, and token. Then, the story's next event is predicted by considering a story as a Markov chain where each element of the chain is sampled from a distribution. The predicted event is translated again into a natural language sentence.

Although RNN was successful in many Seq2Seq problems, they did not reach expectations in story generation, as they failed in many systems to generate coherent stories after few sentences. This results from the fact that a story is a sequence of consistent events that is longer than an RNN can maintain. As Khandelwal et al. [81] showed, RNN's predictions in practice depend on a relatively small part of the previous tokens. Therefore, as the story generation progresses, RNNs lose the connection between the currently generated event and the previous far off events. This affects the consistency and coherence of the generated story.

Inspired by planning-based story generation, Tambwekar et al. [167] proposed a controllable RNN story generator that accepts a given start and end, i.e., a start state and a final state. Then, reinforcement learning is used to guide the RNN toward reaching the given end from the start state. Specifically, they used reward shaping, a method used in reinforcement learning whereby transitional training rewards are used to guide the learning process. After analyzing the story corpus, the reward function was formulated based on two components: the distance component that measures how far the next event is from the given final event, and the story-verb frequency component that estimates how often the next event appears before the given final event throughout the stories in the corpus.

Ammanabrolu et al. [6] used the policy gradient deep reinforcement learner from Tambwekar et al. [167] for events generation. However, their main contribution was to improve the quality of the generated story text to retain all of the event tokens and enhance the interestingness of stories. They argued that a simple language model generates the story text depending on the story corpus and ignores the input story event details that produce semantically unrelated sentences. Therefore, they proposed four event-to-text models: a retrieve-and-edit model, a template filling model, a sequence-to-sequence with Monte Carlo beam decoding model, and a Seq2Seq with a finite state machine decoder. Experimental results showed that an ensemble of the four models outperforms the individual models.

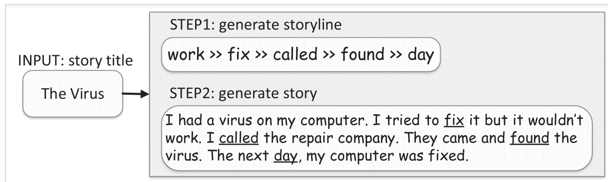


Fig. 4. Hierarchical story generation. Reproduced from Yao et al. [186].

To overcome the drawbacks of RNN in story generation and promote the coherence of generated stories, Fan et al. [54] decomposed story generation into two stages. First, they generated the story premise representing the structure of the story using a convolutional language model. Then, they transformed the premise into a passage of text using a Seq2Seq model. They proposed two mechanisms: a fusion mechanism to improve the relevance between the story and its premise and a self-attention mechanism to model the long-range context. This work inspired some subsequent researches in decomposing story generation into two stages.

Yao et al. [186] proposed another hierarchical story generator that combines plot planning and text generation to generate stories from given titles. During the learning stage, the storyline of each story in the corpus is extracted by pulling out the most important word from each sentence using the RAKE algorithm [147]. Then, two strategies are used to generate stories: the Dynamic Schema and the Static Schema. The Dynamic Schema generates the next word in the storyline and the next sentence in the story at each step. This schema formulates story generation as a content-introducing generation problem. Each word of the generated storyline depends on the context (i.e., the title and previously generated sentences) and the storyline's previous word. However, the Static Schema generates the complete storyline and then translates it into text. This schema formulates story generation as a conditional generation problem where each word of the generated storyline depends on the title and previous word in the storyline. After generating the storyline, a Seq2Seq model is used to translate the storyline into text. The experimental results show that planning the storyline produces better stories in terms of fidelity, coherence, interestingness, and overall user preference. However, the Static Schema outperforms the Dynamic Schema in generating more consistent and coherent stories. Figure 4 shows a sample story generated by this model.

To learn the semantic dependency between sentences in a story, Xu et al. [183] used a reinforcement learning method to learn the most critical phrases of a sentence, called *skeleton*. Based on the skeleton, a Seq2Seq model is trained to generate coherent sentences. However, two factors negatively affected coherence: the input length and the unfamiliarity of input. Like Yao et al. [186], Chen et al. [40] generated outlines as an intermediate step before generating stories. First, they used an off-the-shelf text summarizer to generate high-level plots from the training corpus. Then, they used the natural language summaries to pre-train a planning model on how to generate outlines. A structured Seq2Seq model is used to generate a story given a title and an outline. Although this system outperformed similar previous systems [54, 183, 186], the authors concluded that more powerful mechanisms are needed to improve the coherence at the story level.

Zhai et al. [188] proposed a hybrid model that can generate coherent stories from a small corpus. Their model consists of two modules: an agenda generator that plans the story by sampling a path through a temporal script graph extracted from the story corpus and a neural surface realization module that generates story text conditioned on the story plan. They evaluated the story's global coherence from three different aspects: inclusion, relevance, and order. Araz [10] proposed a transformer neural network for generating stories conditioned on prompts. The generated stories were novel and viable. However, the model generated repetitions and grammatical errors and did not pay much attention to the provided prompts.

Large-scale pre-trained language models have shown high abilities in processing natural languages. Samples of text generated by the GPT-2 model [137] show that these models can generate text comparable to humans' writings. This encouraged researchers to use pre-trained models in story generation. See et al. [157] proposed two models: the pre-trained version of the Fusion model [125] and the smallest version of GPT2 known as GPT2-117 [137]. As in other works [10, 54], the two models were trained to generate stories conditioned on prompts. Overall, they found that the GPT2-117 model is better than the Fusion model in many aspects. Nevertheless, GPT2-117 generated repetitive and under-diverse text when using likelihood-maximizing decoding algorithms. The inefficiency of pre-trained models in story generation was also pointed out by Holtzman et al. [70] when they observed that such models degenerate text by generating text that is bland, incoherent, or gets stuck in repetitive loops.

Guan et al. [64] attributed the lack of information behind the weak performance of pre-trained models. Story generation, as an open-ended generation task, does not provide the model with a gold standard output to compare against, as in other generation tasks such as summarization. This shortage of input obstructs the learning process. Therefore, there is a need for a supporting knowledge source. Thereupon, the author proposed utilizing commonsense knowledge from external knowledge bases to generate good stories. They also used multi-task learning to capture the causal and temporal dependencies between the sentences in a story. Their model generated better stories compared to baseline models in terms of logic and global coherence. Inspired by Guan et al. [64], Xu et al. [184] proposed a controllable story generation framework that allows the dynamical incorporation of commonsense knowledge into the language model. At each generation step, a set of keywords are predicted given the story context. These keywords are used to query the commonsense knowledge base for related concepts. The next sentence of the story is generated by the GPT-2 model conditioned on the story context and the top-ranked retrieved concepts.

Li et al. [94] proposed open-ended causal generation models based on Transformer. They used a causal relations corpus to train a cause model and an effect model. The models generated high-quality and diverse causes and effects. To support diversity, they also developed an approach for disjunctive positive lexical constraints to allow the decoder to select one of a set of provided words or phrases to be included in its output. This approach was employed to choose among different morphological variants of the same lemma.

## 6 KNOWLEDGE SOURCES FOR STORYTELLING

Computational narratives are complex systems that require complex algorithms for the generation process in addition to extensive knowledge. Throughout this survey, we saw a wide range of components being used as knowledge sources for story generation systems, such as author goals, character goals, emotional links, planning rules, and case bases. Although these sources were used successfully in generating stories, they suffer from two shortcomings. First, the high sparsity of knowledge because many story generators develop their own knowledge domains, along with a wide variety of knowledge types. Second, most well known story generators, especially earlier ones, rely mainly on manually crafted knowledge. This knowledge lacks the flexibility to be generalized to other domains, and it costs considerable time and effort to build. To overcome these shortcomings, we believe that it is essential to unify story knowledge sources to enable reusing them by different story generation systems and develop open-domain story generators that can automatically learn domain knowledge. In this section, we will overview the types of knowledge needed by story generators. We will also shed light on open-domain knowledge sources found in the literature: story and semantic relationships corpora, crowdsourcing, and commonsense knowledge.

### 6.1 Types of Knowledge for Story Generation

The knowledge of story generators consists of many extensive interconnected components. Bringsjord and Ferrucci [31] proposed a model for the knowledge needed by computational story generators. Oinonen et al. [121] proposed a similar but more expressive knowledge representation model. We may combine and summarize the two models as follows:

*Thematic knowledge*: Settings of the story are identified such as time, place, and objects. This knowledge should also describe the story world concepts, their properties at a certain point in time, and their relationships.

*Characters knowledge*: A complex and extensive representation of intelligent characters including characters' goals, physical state, personality, and emotional relationships.

*Plot knowledge*: Includes the knowledge needed to construct the story plot, such as agents, events, goals, and actions, and how these components are linked.

*Linguistic knowledge*: Used for representing the story in specific linguistic structures to present it to the reader in natural language.

*Literary knowledge*: Incorporates principles of storytelling in literature designed to increase story interestingness.

*Feedback knowledge*: Effects of story fragments on the user are collected to predict the audience's response to new stories.

A story plot is the core component in story generation. It represents the skeleton of a story, as Bowen [28] said: "Plot is story." Its importance is reflected in computational narratives where most research, covered in Sections 3, 4, and 5, is devoted to extracting and generating event chains. For that reason, we believe that plot knowledge is the most important knowledge among different types of story knowledge. However, despite the long history of automatic story generation, different researchers' plot knowledge was scattered. This shortage was solved by using commonsense knowledge bases (see Section 6.4), and more importantly, story and semantic corpora (see Section 6.2). Nevertheless, most open-domain story generators learn stories as a possible sequence of events without considering other knowledge types.

By comparing human-authored stories, literary scholars found that many stories share a common structure. Based on the observations, they proposed several universal narrative patterns, where new stories can be generated following these plot patterns but with different story worlds and characters [160]. Some of the universal narrative patterns, such as Propp's structure [135], were employed by computer scientists to build structural story generation models (see Section 3). However, these models generated simple sequences of events without extensive thematic and character knowledge. Another direction is Emergent Narrative [11], where the events of a story are driven by characters' goals, beliefs, plans, and interactions with each other [140]. Although this direction is popular in interactive systems and role-playing games, it was used in a few early textual story generators, such as TALE-SPIN [109], where character goals directed story generation. Nevertheless, stories generated by TALE-SPIN were criticized as being uninteresting [32]. To generate more exciting stories, both Virtual Storyteller [169] and FABULIST [141] created more complex characters and guided character planning by story world knowledge and a story director.

MEXICA [129] guides story generation by constraints rather than explicit goals. In its context constraints, a story world context is created for each character to record emotional links between characters, situations that put characters at risk, and changes in the physical position of a character in the story world. The emotional links include a range of values for brotherly love and amorous love in addition to other user-defined emotions. A more complex character representation is proposed by UNIVERSE [87, 88], which follows a top-down approach where a plot is generated based on author goals rather than character goals. However, to generate coherent stories with believable characters, the system motivates character actions to pursue author goals by generating an event, a new character, or a consistent relationship with characters' traits, relationships, and past events. To achieve this, a substantial set of characters is constructed before generating a story and updated after each event. For each character, a Person Frame is created to include character name, physical and personality traits, goals, interpersonal relationships, marriages, and history of events.

Although thematic and character generation represents the right direction for story generation, this direction is understudied by ML researchers. As we saw in Section 5, most ML models' focus was to learn story events and the causal relations between them. Some models generated stories as a language modeling task using pre-trained language models. We believe that exciting stories can be generated by learning and generating high-level plot abstractions with complex story world knowledge, including thematic and character knowledge. This direction can highly benefit from related fields such as game design and interactive systems.

Linguistic knowledge is used to convert a story from its structured form into a natural language text. This knowledge may be considered optional for story generation because off-the-shelf surface realizers can be used to perform this function, as in the work of McIntyre and Lapata [108] and Rishes et al. [143]. Pre-trained language models can also be used to generate well-formed story text given a story context. See et al. [157] found that the smallest version of GPT2 [137] is a better story generator compared to the neural story generation model proposed by Fan et al. [54] in three properties: it conditions more firmly on the story context, it generates more contentful text, and it is more sensitive to correct ordering of events.

Human evaluation is widely used to assess story quality (see Section 8). Consequently, human feedback can make an important factor in improving and predicting the quality of generated stories. Wang et al. [177] used users' feedback from social media to train a model that predicts feedback on newly generated stories. Delatorre et al. [48] controlled the suspense level of generated stories using users' feedback on words' emotional effect. Sagarkar et al. [151] trained a story completion scorer that evaluates a story continuation based on three criteria: overall quality, relevance, and interestingness. The scores predicted by their model strongly correlated with human evaluations.

Literary theories have always influenced automatic story generation as a creative act rooted in human literary. In Section 7, we will discuss in detail how automatic story generators adopted literary theories to increase stories interestingness. Nonetheless, these theories affected the design of the story generation process and were not part of the generation system knowledge base. Therefore, it is safe to say, the system designer requires literary knowledge.

### 6.2 Story and Semantic Relationships Corpora

The recent years witnessed a direction to use ML for modeling computational story generators (see Section 5). These models depend mainly on data for directing the learning process. Therefore, several story datasets were used for developing story generation systems. Table 5 lists the story datasets described in the literature.

In computational narratives, it is crucial to understand and thus generate related events. Causality is one of the primary semantic relationships between events where an event results in another event to happen or hold [115]. The cause-effect relation implicitly implies a temporal order of events. Therefore, temporal and causal relations are closely related. Both have gained much attention in the research community, and several corpora were annotated with these relations. We believe that story generation systems can significantly benefit from research on events

Table 5. Story Generation Datasets

| Dataset | No. of Stories | URL Link |
|---|---|---|
| Gigaword corpus | 1M | https://catalog.ldc.upenn.edu |
| Andrew Lang fairy tale corpus | $>437$ | http://www.mythfolklore.net/andrewlang |
| ROCStories | 98,161 | http://cs.rochester.edu/nlp/rocstories |
| Children's Book Test | 607,627 | https://research.fb.com/downloads/babi |
| Text Annotations from VIST | 41,300 | http://visionandlanguage.net/VIST |
| DUC 2002 | 1,134 | https://www-nlpir.nist.gov/projects/duc/data/2002_data.html |
| WritingPrompts | $\sim 300,000$ | https://www.github.com/pytorch/fairseq |
| CMU movie summary corpus | 42,306 | http://www.cs.cmu.edu/~ark/personas/ |
| Wikipedia's movie plots* | 42,170 | https://dumps.wikimedia.org/enwiki |
| Sci-fi TV show plot summaries | 2,276 | https://drive.google.com/drive/folders/1A5RYjrj9FZsrBtyTr45-fnYWKZX1e7KA |
| STORIUM | 5,743 | https://storium.cs.umass.edu |

*This corpus is the same as the CMU movie summary corpus, just cleaned up a little (hence the fewer stories).

Table 6. Causal and Temporal Relations Corpora as Knowledge Sources for Story Generators

| | Relation | | |
|---|---|---|---|
| Corpus | Casual | Temporal | URL Link |
| CaTeRS [115] | Yes | Yes | http://cs.rochester.edu/nlp/rocstories/CaTeRS |
| TCR [118] | Yes | Yes | http://cogcomp.org/page/publication_view/835 |
| ESC [35] | Yes | Yes | https://github.com/cltl/EventStoryLine.git |
| BECauSE [52] | Yes | No | https://github.com/duncanka/BECauSE |
| CATENA [112] | Yes | Yes | https://github.com/paramitamirza/CATENA |
| Causal-TimeBank [111] | Yes | Yes | http://hlt.fbk.eu/technologies/causal-timebank |
| CausalBank [94] | Yes | No | https://nlp.jhu.edu/causalbank |

semantic relations because stories are full of causal and temporal relations, as demonstrated by Mostafazadeh et al. [115]. Thus, generating story events conditioned on previous semantically related events produces stories with rich events structures. Li et al. [94] used causal relations corpus to guide the generation of cause and effect in the story continuations task. Although covering semantic relations systems is beyond this survey's scope, Table 6 lists recent causal and temporal relations corpora as possible knowledge sources.

### 6.3 Crowdsourcing Knowledge

Crowdsourcing is the process of breaking a complex task into multiple smaller ones that can be completed quickly by people without specific training [90]—typically used to save time and money and to ensure the diversity of the collected data. Crowdsourcing generally benefits from Internet users' enormous potential through forums, social media, or paid crowds. In computational narratives, crowdsourcing was one of the earliest approaches to tackle open-domain story generation, where a story can be generated without relying on manually engineered knowledge. In general, crowdsourcing can be used to obtain all types of knowledge needed by storytelling systems. In addition, it can be used to evaluate generated stories.

SCHEHERAZADE [90] was the first story generation model to use crowdsourcing. First, scripts were collected from non-expert storytellers to build a corpus of a narrative. A plot graph was then built from this corpus, where a story can be generated by sampling the plot graph. Mostafazadeh et al. [114] crowdsourced ROCStories, a corpus of short commonsense, everyday stories that have three main characteristics: realistic, complete with a clear beginning and ending, and do not include anything irrelevant to the core story. These stories are useful for story learning because they are full of stereotypical causal and temporal relations between events.

Fan et al. [54] collected a large dataset of human-written stories by scraping an online forum. Their dataset, the WritingPrompts, consists of medium-length stories paired with short prompts. The prompts were used to inspire story writers. Therefore, this dataset can be used to train story generators to generate stories conditioned on an associated prompt. STORIUM [4] is a story dataset collected from an online collaborative storytelling platform. Each story is broken into discourse-level scene entries annotated with narrative elements, such as character goals or abilities. These annotations are useful for conditioning language models.

Due to the limitations of automated evaluation metrics for computational narratives, subjective evaluation is widely accepted. As humans are involved in the evaluation process, crowdsourcing represents a suitable method for collecting human ratings. Yao et al. [186] employed paid crowds to evaluate stories generated by their system. To ensure evaluation quality, they applied qualification filters to choose evaluation participants. A similar evaluation approach was used by Fan et al. [55] to evaluate their system's output. In contrast, Wang et al. [177] used social media crowdsourced ratings as training data for their system to predict the quality of generated stories. Similarly, Sagarkar et al. [151] crowdsourced annotations for the output of story continuation systems along with several criteria. Then, the annotations were used to train a model to predict the quality of generated stories.

### 6.4 Commonsense Knowledge

Commonsense knowledge refers to beliefs or propositions that appear to be obvious to most people, without dependence on specific esoteric knowledge [73]. This knowledge helps people make assumptions and infer facts about the world that are not explicitly mentioned. Researchers attempted to provide computers with commonsense knowledge bases that simulate human beings' background knowledge, such as synonyms, locations, consequences, and motivations. Examples of commonsense knowledge bases are WordNet [45], **Open Mind Common Sense (OMCS)** [161], ConceptNet [98], Event2Mind [138], and ATOMIC [152].

MAKEBELIEVE [97] was the first story generator that employs commonsense knowledge. It generates short fictional stories based on "consequence" relationships extracted from the OMCS Knowledge Base, thereby inferring causal chains of events and actions representing parts of the story plot. It also uses lexical semantics to connect related ideas that are not identical, which improves the creativity of the generated stories.

In Picture Books [162], a story template was combined with "consequence" relationships extracted from semantic ontology to teach children the consequences of disobedience through a sequence of connected events flowing from negative to positive (rule violation to value acquisition). Although their ontology was constructed manually, they adapted its design from ConceptNet. Several subsequent studies extended this work. In Picture Books 2 [8, 9, 123], additional story locations were added, such as grocery stores and classrooms. In addition, characters were embodied with traits to enhance their believability and direct the system to select the story theme. The ontology structure was manually revised and populated with activities, concepts, and conditions needed for triggering events in the story world. Ontology relationships were increased to enable the system to generate more flexible stories, and a multi-agent planner was used to generate stories.

Soo et al. [164] constructed plot elements by extracting the causal relationships from ConceptNet in the form of Concept-Relationship-Concept triples. The related concepts were then classified based on their semantic and syntactic phrases into action, event, perception, internal element, and goal to determine the type of causal links connecting them. After constructing the plot elements, a **constrained Monte Carlo Tree Search (cMCTS)** algorithm was applied to generate a story plot from the plot elements based on the user's initial state, goal state, and desirable story length. To avoid a cyclic causal sequence, cMCTS removes redundant concepts or semantically similar concepts from the story plot. A simple English sentence generator is designed to generate semantically interpretable sentences from the story plot to enhance its readability.

Commonsense knowledge is vital for open-ended language generation, which usually requires external knowledge to enrich the limited source information. Guan et al. [64] extended a pre-trained model with external commonsense knowledge. They post-trained the model on the knowledge extracted from ConceptNet and ATOMIC, which improved the generated stories' coherence.

## 7 TOWARD INTERESTING STORIES

Stories are artifacts used to deliver knowledge and entertainment. Compared to other texts, stories should be interesting. They aim to create emotional immersion in their recipients and augment pleasure. According to psychologists, several factors can increase a text's interest, such as coherence, causality, completeness, vividness, unexpectedness, suspense, and complexity [155]. Among the interestingness factors, we believe that coherence and causality are the most important factors. Without them, the story may lose its structure and become interconnected with inconsistent fragments. In contrast, the absence of higher-level aspects, such as suspense and surprise, do not affect the story structure but rather decrease its quality. For that reason, coherence and causality were recognized by story generation systems, specifically planning-based models, where inference is based on the causal connections between events. However, literary theories inspired many researchers to improve the interestingness of their auto-generated stories.

***Story suspense.*** Suspense is a narrative procedure used to increase the audience's interest [168]. Modeling suspense computational is a challenge because there is no single unified theoretical definition of suspense [50]. However, there were many attempts to improve the suspense of auto-generated story plots from different aspects:

- *Conflict* in narrative refers to "the struggle in which the actors are engaged" [163]. According to narratologists [1], conflict is an essential element of exciting stories. It occurs when another event in the story thwarts a character's intention—commonly used to create interest in computational narratives. UNIVERSE [86] presents the conflict as part of its pre-scripted stories. MEXICA [129] uses the conflict and resolution processes to generate interesting stories. CPOCL [179] creates a conflict model and then enforces the generation of conflict in stories by constraining the planner. Song et al. [163] propose a way to generate conflicts by manipulating the story plan's causal links to create inter-personal conflicts.
- *Uncertainty* is included in many definitions of suspense [122]. It refers to the possibility that the events of the story do not turn out according to the audience's expectations [1]. Readers enjoy stories with high uncertainty because of their curiosity [83]. Experiments of Gerrig and Bernardo [59] showed that readers feel suspense when led to believe that the quantity or quality of paths through the hero's problem space has become diminished. This definition of uncertainty is specific enough to be implementable.

  The SUSPENSER system [42] modeled uncertainty based on the definition in the work of Gerrig and Bernardo [59]. It generates all possible plans a protagonist might have and computes the suspense level as the inverse number of successful plans. DRAMATIS [122] also used a reformulation of the model of suspense in their work [59]. It reads a story step-by-step to predict whether the protagonist faces a negative outcome and generates an escape plan to avoid that outcome. It computes the level of suspense as the cost of the escape plan. However, some authors question uncertainty as a factor of suspense.

Burget [33] believes that suspense is a fear emotion about an outcome. This fear can be present even if the outcome is known. Delatorre et al. [50] conducted an experiment where the audience read the same story twice. They found that the level of suspense in the second read increased although the story was known in advance. They concluded that uncertainty affects the readers' emotional response, but it is not a feature of suspense. We believe that as long as certainty has an emotional effect on readers, it is a factor of story interestingness despite the relationship between suspense and uncertainty.

- *Emotions* like empathy and hope are naturally evoked by an engaging story. Kintsch [82] classifies interest into two types: cognitive interest that occurs based on the importance of events to the structural development of the story, and emotional interest that occurs when the story arouses a strong emotional response in the reader. Triggering such a response can be done in many ways. According to structural affect theory [30], a story that follows the dramatic arc structure has a better emotional effect on the audience, consequently engaging them. Delatorre et al. [48] used the emotion of fear to increase story suspense. They adopted the definition by Zillman [192], which links suspense to the reader's fearful apprehension of a story event that threatens a liked protagonist. They selected liked protagonists as targets for feared outcomes and created high degrees of subjective certainty for these outcomes. They also discussed the negative effect of high emotional immersion on more sensitive people. Other systems modeled the character's emotions to produce an emotional impact on the readers.

  MEXICA [129] records emotional links between characters and uses a tension curve to represent these emotions to guide the generation process. As the story proceeds, the interactions between characters affect the emotional links between them. MINSTREL [172] employed emotionally charged scenes to add interest to its generated stories. Mori et al. [113] analyzed the relationship between emotions and story interestingness and concluded that the reader's feelings have a higher impact on story interestingness than the characters' emotions.

***Discourse.*** Genette [58] proposed a narrative model with three elements: story, narrative discourse, and the act of narration. Whereas a story is defined as a temporal sequence of events, a discourse is related to the various ways in which a story can be edited and narrated [119]. According to Genette [58], discourse consists of several components such as temporal order alteration, distance, and focalization. These components aim to produce different cognitive and emotional responses of the reader [69]. Bae and Young [15, 16] proposed Prevoyant, a computational story generator that arouses the reader's surprise using two temporal narrative devices: flashback and foreshadowing. Flashback describes some past events related to the present, whereas, foreshadowing gives hints about a future event in a way that makes it difficult for the reader to recognize its meaning until the event happens.

Winer et al. [182] defined structural properties of discourse to provide a basis to reason about the temporal order of events in the discourse. Focalization is another discourse component where a story is viewed from a restricted perspective, such as from one of the story characters [119]. Bae et al. [14] proposed a computational model of focalization using a planning-based approach where each focal character has a different plan library.

Fabula Tales [100] is a story generator that implements narratological variations such as focalization, character voice, and direct vs. indirect speech. Their model implements discourse independently from the story. However, as the distance is a subcategory of discourse theory, it means the quantity of the presence in a story where showing formless distance than telling [58]. Ogata and Yamakage [120] modeled distance in their story generator by compressing narrative information for a longer distance and including internal monologue of characters for a shorter distance.

***Characters.*** The complexity of characters adds to the interestingness of stories. There are two possible approaches. Either create complex characters or create complicated relations between characters. UNIVERSE [86] was one of the first systems that gave attention to character creation. It used a complex structure to represent characters and enhanced interest by creating unusual but believable characters. It stored characters' histories, family relations, and interpersonal relationships. Moreover, it kept track of ongoing plots to update the stored information. Based on elective affinities theory in the work of Goethe [62], Méndez et al. [110] modeled four levels of affinity between characters to represent human relations. Then, it simulates the correlation between characters' interactions and their affinity levels.

***Dialogue.*** Rendering a story as dialogue produces more engagement to the listener [27]. Bowden et al. [27] presented algorithms for converting a deep representation of a story into dialogic storytelling. Their system is capable of telling a story in different settings to different audiences. Petac et al. [130] proposed a system that delivers narrative content through the conversation between these agents. Their system generates rich and engaging dialogues between characters from a formalized plot description. Xu et al. [185] investigated the potential of generating more stylistic dialogues within the context of narratives. They used LSTM-RNN to generate dialogues based on the relations between utterances and narrative actions.

***Narrative text.*** Bradley and Lang [29] devised **Affective Norms for English Words (ANEW)** to provide a set of normative emotional ratings for a large number of words in the English language. ANEW is a corpus of 1,034 non-contextualized words that were rated in terms of pleasure, arousal, and dominance. Its experiment has been replicated for other languages such as French, Finnish, Dutch, Portuguese, or Italian [49]. Delatorre et al. [48] proposed a system that uses ANEW words as effective terms that do not change the plot but rather decorate the plot to the required suspense intensity by controlling the audience's emotional effect. Theoretically, text complexity and quality raise its interestingness [155]. Some researchers used **natural language generation (NLG)** techniques to enhance story text's quality and complexity [2, 101, 103].

Although computational narratives modeled different aspects of story interestingness, there is room for improvement by either using different approaches for implementing the mentioned factors or modeling other interesting factors. In general, research on improving story interestingness can benefit from narrative theories and the advances in other related fields such as gaming and interactive narrators, where more progress was achieved in terms of dialogue, discourse, and conflict, as we noticed while preparing this survey. Further, the current advances in **deep learning (DL)** text generation will increase the quality and control the complexity of narrative text; thereby, story interestingness will increase.

## 8 STORY EVALUATION

Automatic evaluation of stories is very important for story generation. Its importance is not only for evaluating the generated story but also for directing the generation process. However, although story generation systems have undergone many improvements and can generate acceptable results, story evaluation falls behind and is still considered an ongoing problem. Compared to other AI models, story generation adds subjectivity, diversity of evaluation criteria, the high dimensionality of story components, and hence a vast space of possible stories. Therefore, objective measures used for traditional AI models such as completeness and optimality are not applicable for story evaluation. It is also worth mentioning that most computationally generated stories are evaluated based on quality but not creativity. Some researchers argue that assessing computational creativity is impossible because creativity has not been sufficiently studied in human creativity research [79]. However, although human creativity is concerned with h-creativity, i.e., historical creativity [25], it is satisfactory to measure the p-creativity of computers, i.e., psychological creativity where the produced output differs from examples previously seen by the computer.

Many story generation systems, including data-driven systems, rely on human judgments to direct the generation process or evaluate generated stories. For story evaluation, human evaluators are asked to rate the generated stories based on different criteria such as consistency, coherence, and interestingness [108, 129]. Another approach for story subjective evaluation is to ask evaluators to edit generated stories and calculate the story quality measure as the distance between the edit and the original story [90, 144]. Regardless of the wide usage of human evaluation, it suffers from being inflexible, time/effort consuming, and subjective, and it has no gold standard for comparing different story generation systems. Human evaluators also use their background knowledge and imagination to complete inconsistent stories giving them a higher rating than they deserve [126].

The narrative cloze is a sequence of story events from which one event is removed. Chambers and Jurafsky [36] proposed the narrative cloze test to evaluate unsupervised script learning and generation. This type of learning measures a system's ability to predict the missing event by generating a ranked list of guesses for the missing event based on seen events. After generating the predictions list, the system is evaluated using the following metrics.

*Average rank.* Here, the position of the correct event $c$ is averaged over all events for the final score [36]:

$$\frac{1}{|C|}\sum_{c\in C}\text{rank}(c), \tag{4}$$

where $C$ is the full set of events consisting of $|C|$ events.

*Recall@N.* This metric is calculated as the fraction of partial scripts where the missing event is ranked $N$ or less in the list of guesses [76]:

$$\frac{1}{|C|}\,|\{c \mid c \in C \wedge \text{rank}(c) \leq N\}|. \tag{5}$$

*Accuracy.* Previous metrics evaluate guessed events as atomic events where an event is assumed correct if it completely matches the held-out event in all of its attributes. Partially correct guesses are considered wrong guesses. In contrast, accuracy evaluates each attribute of the event separately. Therefore, it is regarded as a more practical and robust metric. For each partial script, the top guess's accuracy is calculated as a fraction of correctly guessed event attributes over the total number of attributes. This value is averaged over all of the test set [131].

The SCT, proposed by Mostafazadeh et al. [114], is based on the narrative cloze test but is designed for supervised learning approaches. It transforms story understanding/generation into a classification problem by giving a system a four-sentence story and two alternatives for the fifth sentence labeled as a "right ending" and "wrong ending." The system performance is measured based on its ability to choose the correct ending for each story. To enable the SCT test, they created the ROCStories Corpora. It is a benchmark corpus of 50,000 crowdsourced five-sentence stories

Table 7. SCT Example from the ROCStories Corpora

| Context | Right Ending | Wrong Ending |
|---|---|---|
| "Karen was assigned a roommate her first year of college. Her roommate asked her to go to a nearby city for a concert. Karen agreed happily. The show was absolutely exhilarating." | Karen became good friends with her roommate. | Karen hated her roommate. |

that capture a rich set of causal and temporal relationships between events (Table 7). An improved version of SCT dataset was crowdsourced by Sharma et al. [158].

Granroth-Wilding and Clark [63] proposed Multiple Choice Narrative Cloze (MCNC), a narrative cloze test where the system is given five randomly ordered events to choose the missing event from. This enables a system to use richer information related to the context and the list of choices. It is also better at comparing different story generation systems.

Once a reference exists for comparing the generated story components, several metrics can be used to evaluate the story generators. Here we give a summary of the evaluation metrics used in the literature.

***Statistical models.*** These models predict story events based on several statistical criteria:

*N-gram overlap*: As in other NLG tasks, story generation quality can be assessed by calculating the *n*-gram overlap between the predicted and expected events. This includes metrics such as BLEU, METEOR, CIDEr, and ROUGE. However, BLEU is the most widely used metric for story generation, e.g., see [7, 65, 105, 183, 186].

*Perplexity*: Perplexity is an evaluation metric commonly used to assess the quality of language models. It measures the prediction ability of a model given the previous context where lower perplexity indicates better prediction accuracy. Perplexity [6] is defined as follows:

$$\text{Perplexity} = 2^{-\sum_x p(x)\log_2 p(x)}, \tag{6}$$

where $x$ is a token in the text, and,

$$p(x) = \frac{\text{count}(x)}{\sum_{y \in Y} \text{count}(y)}, \tag{7}$$

where $Y$ is the vocabulary. Many ML models use perplexity, e.g., [7, 54, 105].

*Pointwise Mutual Information*: PMI is used when an event is selected from several alternatives. It relies on word co-occurrence counts and chooses the event whose co-referring entity has the highest average PMI score within the story chain (see Section 5.2). Initially proposed by Chambers and Jurafsky [36], others have now adopted it, e.g., [76, 148].

*Embeddings models.* Embeddings models predict the story events based on embeddings, either at the word level or the sentence level. Different embedding-based metrics can be used, including Skip Thoughts Cosine Similarity (STCS), Embedding Average Cosine Similarity (EACS), Vector Extrema Cosine Similarity (VECS), and the Greedy Matching Score (GMS). The Average Maximum Similarity model proposed by Roemmele et al. [146] is a word-level embedding model that calculates the mean of the highest similarity embedding for each word of the ending, then selecting the ending with the highest mean. The Deep Structured Semantic model is another structured embedding model applied by Mostafazadeh [114]

for the SCT. The Conditional Generative Adversarial Networks model was also applied in story generation, where the discriminator is used to choose the correct story ending [92].
*Sentiment analysis models.* These models choose the ending with a sentiment that matches the average sentiment of the context, or the sentiment of the last sentence of the context. It is used by Flor and Somasundaran [56] and Mostafazadeh et al. [114].

Evaluating generated stories against a reference is widely accepted, we believe that this practice has its drawbacks. First, treating story generation as a classification problem may lead to good classifiers that do not understand a story's semantics and may not be creative enough to generate a story. Second, generating a story is a creative process by nature. Although these models aim to enhance the systems' prediction ability, a predictive story is not interesting. Furthermore, there is no single answer that should be considered as the only correct answer. This fact contradicts how these evaluation models work by specifying that the system must choose or otherwise be penalized.

Another evaluation approach would be through the story's linguistic properties. Roemmele et al. [145] used three metrics for story-dependent linguistic evaluation measures: *lexical cohesion*, where words in the generated sentence should be semantically related to the words in the story context; *style matching*, where generated sentences should match the style of the story context; and *entity co-reference*, where referring expressions should be used to refer to previously mentioned entities. However, other story-independent linguistic evaluation measures such as spelling, grammar, and lexical diversity could in many systems be a measure for the NLG component rather than the story generator. Purdy et al. [136] assessed the generated story's quality based on four features: grammaticality, temporal ordering, local contextuality, and narrative productivity. Local contextuality checks whether adjacent sentences preserve context, whereas narrative productivity uses several metrics to evaluate stories' reading ease and lexical complexity.

Customized statistical evaluation measures are also used, e.g., in the work of Kartal et al. [80], where the story's believability is calculated as the product of a manually assigned believability value for each action of the story. Soo et al. [164] used weights of causal links to guide the story generation and added some constraints that should be satisfied. León and Gervás [89] guided the story's generation by three different aspects: accumulation of contributions, the appearance of patterns, and inference. These aspects are evaluated based on the values of 13 variables such as interest, tension, causality, and hypotheses. The diversity evaluation metric proposed by Yao et al. [186] measures inter- and intra-story repetition to reduce the repetition rate and generate more diverse stories.

Automatic evaluation of story interestingness attracted several researchers. Wang et al. [177] collected upvotes from social media as an approximate measure for story quality. They trained a neural model to predict upvotes based on textual regions and the interdependence among regions. Sagarkar et al. [151] crowdsourced interestingness evaluation of stories continuations among other criteria. Similar to Wang et al. [177], they trained a neural model to predict human scores. The predicted scores were comparable to human evaluations.

Based on cognitive theories, Behrooz et al. [21, 22] proposed a model that evaluates story interestingness based on the unexpectedness of story events and the story's ability to generate predictive inference in the reader's mind. For estimating unexpectedness, they used word embedding vectors to find the distance between each vector and the average of all vectors belonging to a story. They also used word embeddings to find cases of foreshadowing in a story, as a common cause of predictive inference. The proposed measures were in line with the human judgment of story interestingness. O'Neill and Riedl [122] evaluated suspense in stories based on a cognitive definition of suspense. They defined an escape plan as the process to avoid a negative outcome for the protagonist. Predicting story suspense starts by reading a story in a discretized symbolic-logic format called *time-slices*. At each time-slice, the suspense level is calculated as the resulting escape plan's cost at that point. As the story proceeds, the change in suspense over time creates the suspense curve, and the overall suspense level of a given story is just the area under the curve.

## 9 DISCUSSION

Automatic story generation has been of interest for many decades. Numerous systems have been developed based on different AI approaches and various psychological theories. Nevertheless, all existing story generators are weak AI systems because their results are not comparable with human-generated stories in terms of creativity, originality, and brevity [75]. Despite the long history of automatic story generation, advances in this field to date are less than expected and suffer from many limitations.

*Dispersion.* Because there is no common domain knowledge or evaluation criteria between story generation systems, it is difficult to compare their performances. A good story may result from well-engineered domain knowledge and not an effective generation system. A good story evaluation may result from personal human opinion since most systems depend on human evaluation. More importantly, standardizing domain knowledge and evaluation metrics helps point out each model's strengths and weaknesses and therefore allows other researchers to enhance previous models and build over each other's efforts. Moreover, this explains the modest advances in the story generation field, although decades have passed since it started. Recent researchers started to reuse story corpora and commonsense knowledge bases. Although many researchers used several metrics, none of the existing corpora or evaluation metrics became a standard.

*Domain knowledge.* Not until the start of this decade did all story generation systems rely on manually crafted domain models, which produce closed-domain stories that cannot be extended to other domains. In addition, the manual engineering of the knowledge costs time and effort and must be performed efficiently. Otherwise, it may lead to over-generation or generate stories that are just a reassembly of source stories used to build the knowledge domain. Designing storytelling knowledge can highly benefit from related research fields such as game design and interactive entertainment. The recent advances in data science, including data acquisition and ML, have encouraged the development of open-domain story generators. Several types of open-domain knowledge were used:

- *Story corpora* either contain short stories or plot summaries or consist of everyday stories that are scripts or procedural expositions rather than artifacts. Although they are sufficient for learning story generation, we believe that a more complex corpus is needed for generating higher-level stories in terms of complexity and interestingness.
- *Crowdsourcing* represents a good technique for collecting data. However, due to the absence of standardization, crowdsourcing adds to the dispersion problem.
- *Commonsense knowledge* is a good source of open-domain data. However, we believe that it is a complementary data source that can be used to extend the knowledge of both planning-based and ML story generation systems.
- *Events semantic relations* are critical in stories. But semantic relations corpora usage is extremely limited. Like commonsense knowledge, we believe that semantic relations corpora can be used to extend the story generator's knowledge.

*Seq2Seq models.* These are open-domain story generators that use ML to create stories. Nevertheless, these models' performance was below expectations and has fallen behind many story generators with manually crafted knowledge. Three main limitations caused the modest performance of Seq2Seq story models:

- *Losing consistency*: As the story generation progresses, the logical connections between the currently generated event and previous far off events are lost. Therefore, the story's overall consistency cannot be maintained, as we observed in many systems. However, this limitation was overcome by recent hierarchical models that decompose story generation into a multi-level problem rather than word-level generation, e.g., [54, 55, 183, 186], and by applying reinforcement learning for plot generation, e.g., [167].
- *Repetitive words*: Like other NLG systems, Seq2Seq models tend to generate repetitive words in the generated text. Zhao et al. [189] solved this problem using the coverage model where a coverage vector is maintained to keep track of the attention history to adjust future attention. Yao et al. [186] also tackled this problem by applying heuristics to forbid any word to appear twice when generating a storyline, which indirectly reduces repetition in stories. Furthermore, Fan et al. [54] found that using a top $k$ random sampling scheme reduces repetitive text. Their approach was inspired by Holtzman et al. [71], where a committee of specialized discriminators is trained to address the limitations of the RNN generator.
- *OOV problem*: To reduce the computational expenses, most NLG systems pre-define a word shortlist that contains the top $k$ most frequent words in the training corpus. All other words are replaced by a unique token, called *UNK* (UNknown Word). This practice causes the loss of some information, making it challenging to model rare and unfamiliar words, commonly known as the OOV problem. The pointer mechanism was proposed to tackle this issue in story generation [189].

*Pre-trained language models.* The emergence of pre-trained language models was a giant leap forward in different fields of NLP research. However, these models did not achieve similar success in NLG tasks. They still suffer from repetition, logic conflicts, and lack of long-range coherence [64]. In addition, they cannot manage commonsense inference effectively [176]. Nevertheless, there were multiple attempts to employ these models in story generation and to overcome their shortages. In addition to the questioned efficiency of pre-trained language models in NLG tasks in general, we believe that one of the main reasons for the less-than-expected performance of these models in story generation is that they consider stories as textual pieces, whereas stories are much more complicated structures. Decomposing story generation into subtasks and combining DL models with other generation approaches or extending them with knowledge resources can generate better stories, e.g., [64, 94].

*Story interestingness.* Story consistency and story structure are essential recipes for an interesting story. Whereas story structure is affected by the key steps that formulate a successful story, story consistency is concerned with story believability and whether events and character actions seem logical in the context of the generated story. For an exciting story, both its consistency and structure should be considered. However, this is not the case in most existing story generators. Structural models and some of the planning-based models focus on story structure, whereas most data-driven models and planning-based models focus on story consistency. Only a few systems aim to balance the two properties, e.g. [142, 162, 169]. Many hierarchical story generation models were proposed where a storyline is generated first and then the story is generated from the storyline. Although none of these models have attempted to evaluate generated stories' interestingness, we believe that such models represent a foundation for generating interesting, structured, and consistent stories.

The hierarchical decomposition allows the model to evaluate story structure by evaluating the storyline. Then, the consistency of the story can be controlled at the generation phase. In addition, story interestingness can be enhanced by incorporating approaches inspired by cognitive science and literature. Many of the implemented interestingness methods, reviewed in Section 7, can be easily added to any story generation system.

*Objective evaluation.* From our perspective, one of the main reasons for generating low-quality stories is the absence of useful automatic story evaluation metrics. This is especially true in ML approaches, where evaluation is crucial for guiding the learning process. Early ML story generators used common NLG metrics such as BLEU, ROUGE, and perplexity. These metrics are not suitable for an open-domain generation because they require a gold standard to compare the generated text against, which also conflicts with story generation's creative nature. They also do not correlate with human judgments [96]. Several approaches were used for evaluating generated stories. However, more recent models emulated human judgments by using evaluation criteria such as causality and suspense. It appears that more cognitive science based interestingness theories must be adopted to implement automatic story evaluation algorithms.

## 10 CONCLUSION AND FUTURE DIRECTIONS

Although automatic story generation is a long-standing computational problem, the recent advances in ML are expected to accelerate this field's development. It is worth mentioning that there is a renewed interest in automatic story generation. We believe that as of this writing, there is much ongoing research on this topic. Therefore, it is worth pointing out the limitations of our survey and propose some future research directions.

*Decomposition.* Stories are complex artifacts that have a large number of features. When generating a story, several attributes must be considered: author goals, characters interactions, consistency, structure, suspense, and story text. Therefore, we believe that it would be more efficient to decompose story generation into multiple interacting modules, each of which performs a subtask. Decomposition also includes generating stories in multiple steps, such as separating plot generation from text generation.

*Deep learning.* DL's ability to solve high-dimensional problems fits the high dimensionality of story generation. We have discussed in this survey how recent hierarchical models were able to generate consistent stories. Nonetheless, as DL is yet emerging, new deep story generators are expected to overcome the previous models' limitations.

*Hybrid systems.* These are closely related to decomposition. Once story generation is divided into sub-models, different generation approaches can be used for each sub-model. In particular, we believe that combining planning-based techniques and ML approaches can produce good story generators. Furthermore, combining multiple knowledge sources can also enhance the quality and diversity of generated stories.

*Automatic evaluation.* Proposing efficient automatic story evaluation metrics is essential for assessing story models and thus accelerating the development of new models. The need for automatic evaluation arises in open-domain generation systems where no gold standard exists for comparing the generated result. Thus, evaluating the fitness of different alternatives acts as an objective function for directing the learning process. As discussed previously, several metrics were used to evaluate generated stories, including metrics that can be used for open-domain generation. Nonetheless, the need for effective evaluation metrics is still necessary. Evaluation models can also be trained to evaluate stories based on different criteria.

*Benchmarking.* It is essential to unify both the datasets and evaluation metrics to compare the different proposed models. Benchmarking helps to identify each model's strengths and limitations and thus accelerates the development of more effective story generators. Most existing datasets contain storylines or stories that lack a dramatic structure, and hence they present a description of a sequence of daily events rather than an exciting story. This directly affects the interestingness of generated stories.

## ACKNOWLEDGMENTS

The authors are grateful to the anonymous reviewers for their critical and constructive comments that helped improve this presentation.

## REFERENCES

[1] H. Porter Abbott. 2008. *The Cambridge Introduction to Narrative*. Cambridge University Press.
[2] Emily Ahn, Fabrizio Morbini, and Andrew Gordon. 2016. Improving fluency in narrative text generation with grammatical transformations and probabilistic parsing. In *Proceedings of the 9th International Natural Language Generation Conference*. 70–73.
[3] Taisuke Akimoto. 2016. Exploratory approach to the computational modeling of narrative ability for artificial intelligence. *International Journal of Knowledge Engineering* 2, 4 (2016), 170–176.
[4] Nader Akoury, Shufan Wang, Josh Whiting, Stephen Hood, Nanyun Peng, and Mohit Iyyer. 2020. STORIUM: A dataset and platform for human-in-the-loop story generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. 6470–6484.
[5] James Allan, Ron Papka, and Victor Lavrenko. 1998. On-line new event detection and tracking. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 37–45.
[6] Prithviraj Ammanabrolu, Ethan Tien, Wesley Cheung, Zhaochen Luo, William Ma, Lara Martin, and Mark Riedl. 2019. Guided neural language generation for automated storytelling. In *Proceedings of the 2nd Workshop on Storytelling*. 46–55.
[7] Prithviraj Ammanabrolu, Ethan Tien, Wesley Cheung, Zhaochen Luo, William Ma, Lara J. Martin, and Mark Riedl. 2020. Story realization: Expanding plot events into sentences. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*. 7375–7382.
[8] Karen Ang and Ethel Ong. 2011. Enhancing event-based semantics in the ontology of picture books 2. In *Proceedings of the 8th National Natural Language Processing Research Symposium*. 81–84.
[9] Karen Ang and Ethel Ong. 2012. Planning children's stories using agent models. In *Proceedings of the Pacific Rim Knowledge Acquisition Workshop*. 195–208.
[10] Kemal Araz. 2020. *Transformer Neural Networks for Automated Story Generation*. Master's Thesis. Technological University Dublin, Ireland.
[11] Ruth Aylett. 1999. Narrative in virtual environments—Towards emergent narrative. In *Proceedings of the AAAI 1999 Fall Symposium on Narrative Intelligence*. 83–86.
[12] Ruth Aylett, Sandy Louchart, and Allan Weallans. 2011. Research in interactive drama environments, role-play and story-telling. In *Proceedings of the International Conference on Interactive Digital Storytelling*. 1–12.
[13] Ruth Aylett, Marco Vala, Pedro Sequeira, and Ana Paiva. 2007. FearNot! – An emergent narrative approach to virtual dramas for anti-bullying education. In *Proceedings of the International Conference on Virtual Storytelling*. 202–205.
[14] Byung-Chull Bae, Yun-Gyung Cheong, and R. Michael Young. 2011. Toward a computational model of focalization in narrative. In *Proceedings of the 6th International Conference on Foundations of Digital Games*. 313–315.
[15] Byung-Chull Bae and R. Michael Young. 2008. A use of flashback and foreshadowing for surprise arousal in narrative using a plan-based approach. In *Proceedings of the Joint International Conference on Interactive Digital Storytelling*. 156–167.
[16] Byung-Chull Bae and R. Michael Young. 2013. A computational model of narrative generation for surprise arousal. *IEEE Transactions on Computational Intelligence and AI in Games* 6, 2 (2013), 131–143.
[17] Paul Bailey. 1999. Searching for storiness: Story-generation from a reader's perspective. In *Working Notes of the Narrative Intelligence Symposium*. 157–164.
[18] Niranjan Balasubramanian, Stephen Soderland, Mausam, and Oren Etzioni. 2012. Rel-grams: A probabilistic model of relations in text. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*. 101–105.

[19] Niranjan Balasubramanian, Stephen Soderland, Mausam, and Oren Etzioni. 2013. Generating coherent event schemas at scale. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 1721–1731.
[20] Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics* 34, 1 (2008), 1–34.
[21] Morteza Behrooz, Justus Robertson, and Arnav Jhala. 2019. Investigating the use of word embeddings to estimate cognitive interest in stories. In *Proceedings of the 41st Annual Conference of the Cognitive Science Society (CogSci'19)*, A. K. Goel, C.M. Seifert, and C. Freksa (Eds.). Cognitive Science Society, 1388–1394.
[22] Morteza Behrooz, Justus Robertson, and Arnav Jhala. 2019. Story quality as a matter of perception: Using word embeddings to estimate cognitive interest. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, Vol. 15. 3–9.
[23] John Black and Gordon Bower. 1980. Story understanding as problem-solving. *Poetics* 9, 1–3 (1980), 223–250.
[24] John Black and Robert Wilensky. 1979. An evaluation of story grammars. *Cognitive Science* 3, 3 (1979), 213–229.
[25] Margaret Boden. 2004. *The Creative Mind: Myths and Mechanisms*. Routledge.
[26] Margaret Boden. 2009. Computer models of creativity. *AI Magazine* 30, 3 (2009), 23.
[27] Kevin Bowden, Grace Lin, Lena Reed, Jean Tree, and Marilyn Walker. 2016. M2D: Monolog to dialog generation for conversational story telling. In *Proceedings of the International Conference on Interactive Digital Storytelling*. 12–24.
[28] Elizabeth Bowen. 1945. Notes on writing a novel. *Orion* 2 (1945), 18–30.
[29] Margaret Bradley and Peter Lang. 1999. *Affective Norms for English Words (ANEW): Instruction Manual and Affective Ratings*. Technical Report C-1. Center for Research in Psychophysiology, University of Florida.
[30] William Brewer and Edward Lichtenstein. 1982. Stories are to entertain: A structural-affect theory of stories. *Journal of Pragmatics* 6, 5–6 (1982), 473–486.
[31] Selmer Bringsjord and David Ferrucci. 1999. *Artificial Intelligence and Literary Creativity: Inside the Mind of Brutus, a Storytelling Machine*. Psychology Press.
[32] Selmer Bringsjord and Dave Ferrucci. 1999. BRUTUS and the narrational case against church's thesis. In *Proceedings of the AAAI 1999 Fall Symposium on Narrative Intelligence*. 105–111.
[33] Martin Burget. 2013. *Works of Alfred Hitchcock: An Analysis*. Master's Thesis. Filozofická fakulta, Masarykova Univerzita, Brno, Czech Republic.
[34] Sandra Carberry. 2001. Techniques for plan recognition. *User Modeling and User-Adapted Interaction* 11, 1–2 (2001), 31–48.
[35] Tommaso Caselli and Piek Vossen. 2017. The event storyline corpus: A new benchmark for causal and temporal relation extraction. In *Proceedings of the Events and Stories in the News Workshop*. 77–86.
[36] Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of the 46th Association for Computational Linguistics Conference: Human Language Technologies*. 789–797.
[37] Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, Vol. 2. 602–610.
[38] Nathanael Chambers, Shan Wang, and Dan Jurafsky. 2007. Classifying temporal relations between events. In *Proceedings of the 45th Annual Meeting of the ACL: Interactive Poster and Demonstration Sessions*. 173–176.
[39] Snigdha Chaturvedi, Haoruo Peng, and Dan Roth. 2017. Story comprehension for predicting what happens next. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 1603–1614.
[40] Gang Chen, Yang Liu, Huanbo Luan, Meng Zhang, Qun Liu, and Maosong Sun. 2020. Learning to generate explainable plots for neural story generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2020), 585–593.
[41] Jiaao Chen, Jianshu Chen, and Zhou Yu. 2019. Incorporating structured commonsense knowledge in story completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 6244–6251.
[42] Yun-Gyung Cheong and R. Michael Young. 2014. Suspenser: A story generation system for suspense. *IEEE Transactions on Computational Intelligence and AI in Games* 7, 1 (2014), 39–52.
[43] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of 2014 Conference on Empirical Methods in Natural Language Processing*. 1724–1734.
[44] YunSeok Choi, SuAh Kim, and Jee-Hyong Lee. 2016. Recurrent neural network for storytelling. In *Proceedings of the 2016 Joint 8th International Conference on Soft Computing and Intelligent Systems and 17th International Symposium on Advanced Intelligent Systems*. 841–845.
[45] Fellbaum Christiane (Ed.). 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.
[46] William Cook. 2011. *Plotto: The Master Book of All Plots*. Tin House Books.
[47] Natlie Dehn. 1981. Story generation after TALE-SPIN. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, Vol. 1. 16–18.

[48] Pablo Delatorre, Barbara Arfe, Pablo Gervás, and Manuel Palomo-Duarte. 2016. A component-based architecture for suspense modelling. In *Proceedings of the 3rd AISB Symposium on Computational Creativity*. 32–39.

[49] Pablo Delatorre, Carlos León, Pablo Gervás, and Manuel Palomo-Duarte. 2017. A computational model of the cognitive impact of decorative elements on the perception of suspense. *Connection Science* 29, 4 (2017), 295–331.

[50] Pablo Delatorre, Carlos León, Alberto Salguero, Manuel Palomo-Duarte, and Pablo Gervás. 2018. Confronting a paradox: A new perspective of the impact of uncertainty in suspense. *Frontiers in Psychology* 9 (2018), 1392.

[51] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Vol. 1. 4171–4186.

[52] Jesse Dunietz, Lori Levin, and Jaime G. Carbonell. 2017. The BECauSE corpus 2.0: Annotating causality and overlapping relations. In *Proceedings of the 11th Linguistic Annotation Workshop*. 95–104.

[53] Markus Eger, Colin Potts, Camille Barot, and R. Michael Young. 2015. Plotter: Operationalizing the master book of all plots. In *Proceedings of the Conference on Intelligent Narrative Technologies and Social Believability in Games*. 30–33.

[54] Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 889–898.

[55] Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2650–2660.

[56] Michael Flor and Swapna Somasundaran. 2017. Sentiment analysis and lexical cohesion for the story cloze task. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential, and Discourse-level Semantics*. 62–67.

[57] Gustav Freytag. 1968. *The Technique of the Drama: An Exposition of Dramatic Composition and Art*. B. Blom, New York, NY.

[58] Gérard Genette. 1983. *Narrative Discourse: An Essay in Method*. Vol. 3. Cornell University Press.

[59] Richard Gerrig and Allan Bernardo. 1994. Readers as problem-solvers in the experience of suspense. *Poetics* 22, 6 (1994), 459–472.

[60] Pablo Gervás. 2009. Computational approaches to storytelling and creativity. *AI Magazine* 30, 3 (2009), 49.

[61] Pablo Gervás, Belén Díaz-Agudo, Federico Peinado, and Raquel Hervás. 2004. Story plot generation based on CBR. In *Proceedings of the International Conference on Innovative Techniques and Applications of Artificial Intelligence*. 33–46.

[62] Johann Goethe. 1971. *Elective Affinities*. Penguin UK.

[63] Mark Granroth-Wilding and Stephen Clark. 2016. What happens next? Event prediction using a compositional neural network model. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*.

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

[65] Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, Vol. 33. 6473–6480.

[66] B. Harrison, C. Purdy, and M. Riedl. 2017. Toward automated story generation with Markov chain Monte Carlo methods and deep neural networks. In *Proceedings of the 13th Artificial Intelligence and Interactive Digital Entertainment Conference*. 191–197.

[67] Ken Hartsook, Alexander Zook, Sauvik Das, and Mark Riedl. 2011. Toward supporting stories with procedurally generated game worlds. In *Proceedings of the IEEE Conference on Computational Intelligence and Games*. 297–304.

[68] Patrik Haslum. 2012. Narrative planning: Compilations to classical planning. *Journal of Artificial Intelligence Research* 44 (2012), 383–395.

[69] Hans Hoeken and Mario van Vliet. 2000. Suspense, curiosity, and surprise: How discourse structure influences the affective and cognitive processing of a story. *Poetics* 27, 4 (2000), 277–286.

[70] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *Proceedings of the International Conference on Learning Representations*.

[71] Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 1638–1649.

[72] Linmei Hu, Juanzi Li, Liqiang Nie, Xiao-Li Li, and Chao Shao. 2017. What happens next? Future subevent prediction using contextual hierarchical LSTM. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*.

[73] Sheng-Hao Hung, Chia-Hung Lin, and Jen-Shin Hong. 2010. Web mining for event-based commonsense knowledge using lexico-syntactic pattern matching and semantic role labeling. *Expert Systems with Applications* 37, 1 (2010), 341–347.

[74] Daphne Ippolito, David Grangier, Douglas Eck, and Chris Callison-Burch. 2020. Toward better storylines with sentence-level language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 7472–7478.

[75] Parag Jain, Priyanka Agrawal, Abhijit Mishra, Mohak Sukhwani, Anirban Laha, and Karthik Sankaranarayanan. 2017. Story generation from sequence of independent short descriptions. In *Proceedings of the SIGKDD Workshop on Machine Learning for Creativity (ML4Creativity'17).*

[76] Bram Jans, Steven Bethard, Ivan Vulić, and Marie Francine Moens. 2012. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics.* 336–344.

[77] A. Jaya and G. V. Uma. 2010. An intelligent system for semi-automatic story generation for kids using ontology. In *Proceedings of the 3rd Annual ACM Bangalore Conference.* Article 8, 6 pages.

[78] Arnav Jhala and R. Michael Young. 2011. Intelligent machinima generation for visual storytelling. In *Artificial Intelligence for Computer Games.* Springer, 151–170.

[79] Anna Jordanous. 2012. A standardised procedure for evaluating creative systems: Computational creativity evaluation based on what it is to be creative. *Cognitive Computation* 4, 3 (2012), 246–279.

[80] Bilal Kartal, John Koenig, and Stephen Guy. 2014. User-driven narrative variation in large story domains using Monte Carlo tree search. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems.* 69–76.

[81] U. Khandelwal, He He, P. Qi, and D. Jurafsky. 2018. Sharp nearby, fuzzy far away: How neural language models use context. In *Proceedings of 56th Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 284–294.

[82] Walter Kintsch. 1980. Learning from text, levels of comprehension, or: Why anyone would read a story anyway. *Poetics* 9, 1–3 (1980), 87–98.

[83] Silvia Knobloch-Westerwick and Caterina Keplinger. 2006. Mystery appeal: Effects of uncertainty and resolution on the enjoyment of mystery. *Media Psychology* 8, 3 (2006), 193–212.

[84] Ben Kybartas and Rafael Bidarra. 2016. A survey on story generation techniques for authoring computational narratives. *IEEE Transactions on Computational Intelligence and AI in Games* 9, 3 (2016), 239–253.

[85] George Lakoff. 1972. Structural complexity in fairy tales. *Study of Man* 1 (1972), 128–150.

[86] Michael Lebowitz. 1983. *Creating a Story-telling Universe.* Technical Report CUCS-055-83. Department of Computer Science, Columbia University. https://doi.org/10.7916/D8RV0WQN

[87] Michael Lebowitz. 1984. Creating characters in a story-telling universe. *Poetics* 13, 3 (1984), 171–194.

[88] Michael Lebowitz. 1985. Story-telling as planning and learning. *Poetics* 14, 6 (1985), 483–502.

[89] Carlos León and Pablo Gervás. 2010. The role of evaluation-driven rejection in the successful exploration of a conceptual space of stories. *Minds and Machines* 20, 4 (2010), 615–634.

[90] Boyang Li, Stephen Lee-Urban, George Johnston, and Mark Riedl. 2013. Story generation with crowdsourced plot graphs. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence.* 598–604.

[91] Qian Li, Ziwei Li, Jin-Mao Wei, Yanhui Gu, Adam Jatowt, and Zhenglu Yang. 2018. A multi-attention based neural network with external knowledge for story ending predicting task. In *Proceedings of the 27th International Conference on Computational Linguistics.* 1754–1762.

[92] Zhongyang Li, Xiao Ding, and Ting Liu. 2018. Generating reasonable and diversified story ending using sequence to sequence model with adversarial training. In *Proceedings of the 27th International Conference on Computational Linguistics.* 1033–1043.

[93] Zhongyang Li, Xiao Ding, and Ting Liu. 2019. Story ending prediction by transferable BERT. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence.* 1800–1806.

[94] Zhongyang Li, Xiao Ding, Ting Liu, J. Edward Hu, and Benjamin Van Durme. 2020. Guided generation of cause and effect. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence.* 3629–3636.

[95] Hongyu Lin, Le Sun, and Xianpei Han. 2017. Reasoning with heterogeneous knowledge for commonsense machine comprehension. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing.* 2032–2043.

[96] Chia-Wei Liu, Ryan Lowe, Iulian Vlad Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT To Evaluate Your Dialogue System: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing.* 2122–2132.

[97] Hugo Liu and Push Singh. 2002. MAKEBELIEVE: Using commonsense knowledge to generate stories. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI'02).* 957–958.

[98] Hugo Liu and Push Singh. 2004. ConceptNet—A practical commonsense reasoning tool-kit. *BT Technology Journal* 22, 4 (2004), 211–226.

[99] Vincenzo Lombardo and Rossana Damiano. 2012. Storytelling on mobile devices for cultural heritage. *New Review of Hypermedia and Multimedia* 18, 1–2 (2012), 11–35.

[100] Stephanie Lukin and Marilyn Walker. 2015. Narrative variations in a virtual storyteller. In *Proceedings of the International Conference on Intelligent Virtual Agents.* 320–331.

[101] Stephanie Lukin and Marilyn Walker. 2019. A narrative sentence planner and structurer for domain independent, parameterizable storytelling. *Dialogue & Discourse* 10, 1 (2019), 34–86.
[102] Shuming Ma and Xu Sun. 2017. A semantic relevance based neural network for text summarization and text simplification. arXiv:1710.02318
[103] Enrique Manjavacas, Folgert Karsdorp, Ben Burtenshaw, and Mike Kestemont. 2017. Synthetic literature: Writing science fiction in a co-creative process. In *Proceedings of the Workshop on Computational Creativity in Natural Language Generation*. 29–37.
[104] Pierre Maranda. 1985. Semiography and artificial intelligence. *International Semiotic Spectrum* 4 (1985), 1–3.
[105] Lara Martin, Prithviraj Ammanabrolu, Xinyu Wang, William Hancock, Shruti Singh, Brent Harrison, and Mark Riedl. 2018. Event representations for automated story generation with deep neural nets. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. 868–875.
[106] Michael Mateas and Phoebe Sengers (Eds.). 2003. *Narrative Intelligence*. John Benjamins Publishing.
[107] Michael Mateas and Andrew Stern. 2003. Integrating plot, character and natural language processing in the interactive drama Façade. In *Proceedings of the 1st International Conference on Technologies for Interactive Digital Storytelling and Entertainment*, Vol. 2.
[108] Neil McIntyre and Mirella Lapata. 2010. Plot induction and evolutionary search for story generation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. 1562–1572.
[109] James Meehan. 1977. TALE-SPIN, an interactive program that writes stories. In *Proceedings of the 5th International Joint Conference on Artificial intelligence*, Vol. 1. 91–98.
[110] Gonzalo Méndez, Pablo Gervás, and Carlos León. 2016. On the use of character affinities for story plot generation. In *Knowledge, Information and Creativity Support Systems*. 211–225.
[111] Paramita Mirza and Sara Tonelli. 2014. An analysis of causality between events and its relation to temporal information. In *Proceedings of the 25th International Conference on Computational Linguistics*. 2097–2106.
[112] Paramita Mirza and Sara Tonelli. 2016. Catena: Causal and temporal relation extraction from natural language texts. In *Proceedings of the 26th International Conference on Computational Linguistics*. 64–75.
[113] Yusuke Mori, Hiroaki Yamane, Yoshitaka Ushiku, and Tatsuya Harada. 2019. How narratives move your mind: A corpus of shared-character stories for connecting emotional flow and interestingness. *Information Processing & Management* 56, 5 (2019), 1865–1879.
[114] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 839–849.
[115] Nasrin Mostafazadeh, Alyson Grealish, Nathanael Chambers, James Allen, and Lucy Vanderwende. 2016. CaTeRS: Causal and temporal relation scheme for semantic annotation of event structures. In *Proceedings of the 4th Workshop on Events*. 51–61.
[116] Nasrin Mostafazadeh, Lucy Vanderwende, Wen-Tau Yih, Pushmeet Kohli, and James Allen. 2016. Story cloze evaluator: Vector space representation evaluation by predicting what happens next. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*. 24–29.
[117] Mark Nelson and Michael Mateas. 2005. Search-based drama management in the interactive fiction anchorhead. In *Proceedings of the 1st Artificial Intelligence and Interactive Digital Entertainment Conference*. 99–104.
[118] Qiang Ning, Zhili Feng, Hao Wu, and Dan Roth. 2018. Joint reasoning for temporal and causal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 2278–2288.
[119] Takashi Ogata. 2016. Computational and cognitive approaches to narratology from the perspective of narrative generation. In *Computational and Cognitive Approaches to Narratology*. IGI Global, 1–74.
[120] Takashi Ogata and Sayaka Yamakage. 2004. A computational mechanism of the "distance" in narrative: A trial in the expansion of literary theory. In *Proceedings of the 8th World Multiconference on Systemics, Cybernetics, and Informatics*, Vol. 14. 179–184.
[121] Katri Oinonen, Mariët Theune, Anton Nijholt, and Jasper Uijlings. 2006. Designing a story database for use in automatic story generation. In *Proceedings of the International Conference on Entertainment Computing*. 298–301.
[122] Brian O'Neill and Mark Riedl. 2014. Dramatis: A computational model of suspense. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*. 944–950.
[123] Ethel Ong. 2010. A commonsense knowledge base for generating children's stories. In *Proceedings of the AAAI Fall Symposium Series on Common Sense Knowledge*. 82–87.
[124] TeongJoo Ong and John Leggett. 2004. A genetic algorithm approach to interactive narrative generation. In *Proceedings of the 15th ACM Conference on Hypertext and Hypermedia*. ACM, New York, NY, 181–182.
[125] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. 48–53.

[126] Federico Peinado and Pablo Gervás. 2006. Evaluation of automatic generation of basic stories. *New Generation Computing* 24, 3 (2006), 289–302.
[127] Lyn Pemberton. 1984. *Story Structure: A Narrative Grammar of Nine Chansons de Geste of the Guillaume d'Orange Cycle*. Ph.D. Dissertation. University of Toronto.
[128] Lyn Pemberton. 1989. A modular approach to story generation. In *Proceedings of the 4th Conference of the European Chapter of the Association for Computational Linguistics*. 217–224.
[129] Rafael Perez y Perez and Mike Sharples. 2001. MEXICA: A computer model of a cognitive account of creative writing. *Journal of Experimental and Theoretical Artificial Intelligence* 13, 2 (2001), 119–139.
[130] Andreea-Oana Petac, Anne-Gwenn Bosser, Fred Charles, Pierre De Loor, and Marc Cavazza. 2020. A pragmatics-based model for narrative dialogue generation. In *Proceedings of the 11th International Conference on Computational Creativity*.
[131] Karl Pichotta and Raymond Mooney. 2014. Statistical script learning with multi-argument events. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. 220–229.
[132] Karl Pichotta and Raymond Mooney. 2016. Using sentence-level LSTM language models for script inference. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 279–289.
[133] Karl Pichotta and Raymond J. Mooney. 2016. Learning statistical scripts with LSTM recurrent neural networks. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. 2800–2806.
[134] Georges Polti. 1916. *The Thirty-Six Dramatic Situations,* L. Ray (Trans.). The Writer, Boston, MA. First published in French in 1895.
[135] Vladimir Propp. 1968. *Morphology of the Folktale*. University of Texas Press.
[136] Christopher Purdy, Xinyu Wang, Larry He, and Mark Riedl. 2018. Predicting generated story quality with quantitative measures. In *Proceedings of 14th Conference on Artificial Intelligence and Interactive Digital Entertainment*. 95–101.
[137] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1, 8 (2019), 1–24.
[138] Hannah Rashkin, Maarten Sap, Emily Allaway, Noah A. Smith, and Yejin Choi. 2018. Event2Mind: Commonsense inference on events, intents, and reactions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 463–473.
[139] M. Riedl, A. Stern, D. Dini, and J. Alderman. 2008. Dynamic experience management in virtual worlds for entertainment, education, and training. *International Transactions on Systems Science and Applications* 4, 1 (2008), 23–42.
[140] Mark Riedl and R. Michael Young. 2003. Character-focused narrative generation for execution in virtual worlds. In *Proceedings of the International Conference on Virtual Storytelling*. 47–56.
[141] Mark Riedl and R. Michael Young. 2004. An intent-driven planner for multi-agent story generation. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems*, Vol. 1. 186–193.
[142] Mark Riedl and R. Michael Young. 2010. Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research* 39 (2010), 217–268.
[143] Elena Rishes, Stephanie Lukin, David Elson, and Marilyn Walker. 2013. Generating different story tellings from semantic representations of narrative. In *Proceedings 6th International Conference on Interactive Storytelling*. 192–204.
[144] Melissa Roemmele and Andrew Gordon. 2015. Creative help: A story writing assistant. In *Proceedings of the International Conference on Interactive Digital Storytelling*. 81–92.
[145] Melissa Roemmele, Andrew Gordon, and Reid Swanson. 2017. Evaluating story generation systems using automated linguistic analyses. In *Proceedings of the SIGKDD 2017 Workshop on Machine Learning for Creativity*.
[146] Melissa Roemmele, Sosuke Kobayashi, Naoya Inoue, and Andrew Gordon. 2017. An RNN-based binary classifier for the story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential, and Discourse-level Semantics*. 74–80.
[147] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. In *Text Mining: Applications and Theory*, Michael Berry and Jacob Kogan (Eds.). John Wiley & Sons, New York, NY, 3–20.
[148] Rachel Rudinger, Pushpendre Rastogi, Francis Ferraro, and Benjamin Van Durme. 2015. Script induction as language modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1681–1686.
[149] David E. Rumelhart. 1975. Notes on a schema for stories. In *Representation and Understanding*. Studies in Cognitive Science. Morgan Kaufmann, 211–236.
[150] James Ryan. 2017. Grimes' fairy tales: A 1960s story generator. *Lecture Notes in Computer Science* 10690 (2017), 89–103.
[151] Manasvi Sagarkar, John Wieting, Lifu Tu, and Kevin Gimpel. 2018. Quality signals in generated stories. In *Proceedings of the 7th Joint Conference on Lexical and Computational Semantics*. 192–202.

[152] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*. 3027–3035.

[153] Roger Schank and Robert Abelson. 2013. *Scripts, Plans, Goals, and Understanding: An Inquiry into Human Knowledge Structures*. Psychology Press.

[154] Victoria Schmidt. 2005. *Story Structure Architect*. Penguin.

[155] Gregory Schraw, Terri Flowerday, and Stephen Lehman. 2001. Increasing situational interest in the classroom. *Educational Psychology Review* 13, 3 (2001), 211–224.

[156] Abigail See, Peter Liu, and Christopher Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vol. 1. 1073–1083.

[157] Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D. Manning. 2019. Do massively pre-trained language models make better storytellers? In *Proceedings of the 23rd Conference on Computational Natural Language Learning*. 843–861.

[158] R. Sharma, J. Allen, O. Bakhshandeh, and N. Mostafazadeh. 2018. Tackling the story ending biases in the story cloze test. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, Vol. 2. 752–757.

[159] Mike Sharples. 1996. An account of writing as creative design. In *The Science of Writing: Theories, Methods, Individual Differences and Applications*, C. Michael Levy and Sarah Ransdell (Eds.). Routledge, 127–148.

[160] Manvir Singh. 2019. The sympathetic plot, its psychological origins, and implications for the evolution of fiction. OSF Reprints. http://doi.org/10.31219/osf.io/p8q7a

[161] Push Singh, Thomas Lin, Erik Mueller, Grace Lim, Travell Perkins, and Wan Li Zhu. 2002. Open mind common sense: Knowledge acquisition from the general public. *Lecture Notes in Computer Science* 2519 (2002), 1223–1237.

[162] Candice Solis, Joan Tiffany Siy, Emerald Tabirao, and Ethel Ong. 2009. Planning author and character goals for story generation. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*. 63–70.

[163] Youngrok Song, Hyunju Kim, Taewoo Yoo, Byung-Chull Bae, and Yun-Gyung Cheong. 2020. An intelligent storytelling system for narrative conflict generation and resolution. In *Proceedings of the IEEE Conference on Games*. 192–197.

[164] Von-Wun Soo, Chi-Mou Lee, and Tai-Hsun Chen. 2016. Generate believable causal plots with user preferences using constrained monte carlo tree search. In *Proceedings of the 12th Artificial Intelligence and Interactive Digital Entertainment Conference*.

[165] Ilya Sutskever, Oriol Vinyals, and Quoc Le. 2014. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems* 27 (2014), 3104–3112.

[166] Reid Swanson and Andrew S. Gordon. 2012. Say anything: Using textual case-based reasoning to enable open-domain interactive storytelling. *ACM Transactions on Interactive Intelligent Systems* 2, 3 (2012), 1–35.

[167] Pradyumna Tambwekar, Murtaza Dhuliawala, Lara Martin, Animesh Mehta, Brent Harrison, and Mark Riedl. 2019. Controllable neural story plot generation via reward shaping. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 5982–5988.

[168] Ed S. Tan. 2013. *Emotion and the Structure of Narrative Film: Film as an Emotion Machine*. Routledge.

[169] Mariët Theune, Sander Faas, Anton Nijholt, and Dirk Heylen. 2003. The virtual storyteller: Story creation by intelligent agents. In *Proceedings of the Technologies for Interactive Digital Storytelling and Entertainment Conference*. 204–215.

[170] Perry Thorndyke. 1977. Cognitive structures in comprehension and memory of narrative discourse. *Cognitive Psychology* 9, 1 (1977), 77–110.

[171] Van-Khanh Tran and Minh Le Nguyen. 2017. Natural language generation for spoken dialogue system using RNN encoder-decoder networks. In *Proceedings of the 21st Conference on Computational Natural Language Learning*. 442–451.

[172] Scott Turner. 2014. *The Creative Process: A Computer Model of Storytelling and Creativity*. Psychology Press.

[173] Josep Valls-Vargas. 2013. Narrative extraction, processing and generation for interactive fiction and computer games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, Vol. 9.

[174] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[175] Bingning Wang, Kang Liu, and Jun Zhao. 2017. Conditional generative adversarial networks for commonsense machine comprehension. In *Proceedings of 26th International Joint Conference on Artificial Intelligence*. 4123–4129.

[176] Su Wang, Greg Durrett, and Katrin Erk. 2020. Narrative interpolation for generating and understanding stories. arXiv:2008.07466

[177] Tong Wang, Ping Chen, and Boyang Li. 2017. Predicting the quality of short narratives from social media. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

[178] Tianming Wang and Xiaojun Wan. 2019. T-CVAE: Transformer-based conditioned variational autoencoder for story completion. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 5233–5239.
[179] Stephen Ware and R. Michael Young. 2011. CPOCL: A narrative planner supporting conflict. In *Proceedings of the 7th Artificial Intelligence and Interactive Digital Entertainment Conference*.
[180] Peter Weyhrauch. 1997. *Guiding Interactive Fiction*. Ph.D. Dissertation. Carnegie Mellon University.
[181] Robert Wilensky. 1983. Story grammars versus story points. *Behavioral and Brain Sciences* 6, 4 (1983), 579–591.
[182] David Winer, Adam Amos-Binks, Camille Barot, and R. Michael Young. 2015. Good timing for computational models of narrative discourse. In *Proceedings of the 6th Workshop on Computational Models of Narrative*. 152–156.
[183] Jingjing Xu, Xuancheng Ren, Yi Zhang, Qi Zeng, Xiaoyan Cai, and Xu Sun. 2018. A skeleton-based model for promoting coherence among sentences in narrative story generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 4306–4315.
[184] Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. 2020. MEGATRON-CNTRL: Controllable story generation with external knowledge using large-scale language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 2831–2845.
[185] Weilai Xu, Charlie Hargood, Wen Tang, and Fred Charles. 2018. Towards generating stylistic dialogues for narratives using data-driven approaches. In *Proceedings of the International Conference on Interactive Digital Storytelling*. 462–472.
[186] Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*. 7378–7385.
[187] R. Michael Young, Stephen Ware, Brad Cassell, and Justus Robertson. 2013. Plans and planning in narrative generation: A review of plan-based approaches to the generation of story, discourse and interactivity in narratives. *Sprache und Datenverarbeitung* 37, 1–2 (2013), 41–64.
[188] Fangzhou Zhai, Vera Demberg, Pavel Shkadzko, Wei Shi, and Asad Sayeed. 2019. A hybrid model for globally coherent story generation. In *Proceedings of the 2nd Workshop on Storytelling*. 34–45.
[189] Yan Zhao, Lu Liu, Chunhua Liu, Ruoyao Yang, and Dong Yu. 2018. From plots to endings: A reinforced pointer generator for story ending generation. In *Proceedings of the CCF International Conference on Natural Language Processing and Chinese Computing*. 51–63.
[190] Chenguang Zhu, Michael Zeng, and Xuedong Huang. 2019. Multi-task learning for natural language generation in task-oriented dialogue. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. 1261–1266.
[191] Jichen Zhu and Santiago Ontañón. 2014. Shall I compare thee to another story?—An empirical study of analogy-based story generation. *IEEE Transactions on Computational Intelligence and AI in Games* 6, 2 (2014), 216–227.
[192] Dolf Zillmann. 1996. The psychology of suspense in dramatic exposition. *Suspense: Conceptualizations, Theoretical Analyses, and Empirical Explorations*, P. Vorderer, H. J. Wulff, and M. Friedrichsen (Eds.). LEA's Communication Series. Lawrence Erlbaum Associates, 199–231.

Received April 2020; revised January 2021; accepted February 2021