Inferring the Reader: Guiding Automated Story Generation with Commonsense Reasoning

Xiangyu Peng[♣*] **Siyan Li**[♣*] **Sarah Wiegreffe**[†] **Mark Riedl**[♣]
[♣]Georgia Institute of Technology
[†]Allen Institute for Artificial Intelligence
{xpeng62, sli613}@gatech.edu
wiegreffesarah@gmail.com
riedl@cc.gatech.edu

Abstract

Transformer-based language model approaches to automated story generation currently provide state-of-the-art results. However, they still suffer from plot incoherence when generating narratives over time, and critically lack basic commonsense reasoning. Furthermore, existing methods generally focus only on single-character stories, or fail to track characters at all. To improve the coherence of generated narratives and to expand the scope of character-centric narrative generation, we introduce Commonsense-inference Augmented neural StoryTelling (CAST), [1] a framework for introducing commonsense reasoning into the generation process with the option to model the interaction between multiple characters. We find that our CAST method produces significantly more coherent, on-topic, enjoyable and fluent stories than existing models in both the single-character and two-character settings in three storytelling domains.

1 Introduction

AI storytelling is a crucial component of computational creativity. Humans use storytelling to entertain, share experiences, educate, and to facilitate social bonding (Riedl and Young, 2010). For an intelligent system to be unable to generate a story limits its ability to interact with humans in naturalistic ways (Riedl, 2016). Automated Story Generation, the task of requiring a system to construct a sequence of sentences that can be read and understood as a story, is a grand challenge in AI.

Prior to the advent of neural language models, methods to model the narrative arcs of stories leveraged a variety of statistical techniques to track events and characters (Gervás, 2013, 2014; Ouyang and McKeown, 2015). The dominant approach to story generation today is to use neural language models (Roemmele, 2016; Khalifa et al., 2017; Clark et al., 2018; Martin et al., 2018). When a language model is trained on a corpus of stories, samples from the resulting distribution tend to also be stories. These techniques have improved with the adoption of Transformer-based models, such as GPT-2 (Radford et al., 2019) and GPT-3 (Brown et al., 2020). However, these models are prone to generating repetitive or generic continuations (Holtzman et al., 2019). Furthermore, as the length of the story grows, these models can lose coherence. Other artifacts include new characters being arbitrarily introduced at any time and characters being forgotten. One reason for these phenomena is that language models generate continuations by sampling from a learned distribution $P_\theta(tok_n|tok_{<n})$. Human readers, however, do not

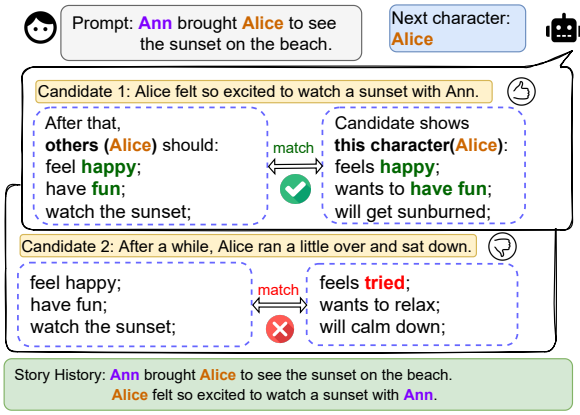

Figure 1: Overview of the CAST system. 1. A text prompt and a specified character start the story generation process. 2. A language model generates candidate continuations (two are shown) with the specified character as the main character. 3. The system infers commensense attributes about the main character from each candidate sentence. 4. If enough inferences from a candidate sentence match those from the prompt sentence, the candidate is added to the story and becomes the new prompt (here, only the first candidate meets this criterion). 5. The process repeats (with the option to specify a new main character) until a story of desired length is generated.

[*]Equal contributions
[1]Code: https://github.com/xiangyu-peng/CAST_public

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 7037 - 7058
December 7-11, 2022

perceive the coherence of a narrative as a function of the likelihood of seeing specific continuations of previous contexts. Statistical sampling from a distribution is not constrained to making logical transitions because the rich relationships that readers make to perceive coherence are not modeled.

Previous attempts to enhance story generation coherence use conditioning on content-relevant features such as plot outlines (Fan et al., 2018; Peng et al., 2018; Rashkin et al., 2020), or character emotional arcs (Brahman and Chaturvedi, 2020). These improve plot coherence through adherence to a manually-given high-level plan. A high level plan can also be automatically generated then decomposed (Yao et al., 2019; Fan et al., 2019; Ammanabrolu et al., 2020b), which elevates the challenges of maintaining coherence to a higher level of abstraction. Neural language models can also be fine-tuned on other signals such as commonsense knowledge or progression rewards (Guan et al., 2020; Tambwekar et al., 2019), which improves the distribution but still relies solely on sampling and the assumption that language models can encode complex story structure in token distributions.

The latent state of neural language models used to generate subsequent story continuations are unlikely to relate to a human reader's mental model of the state of a story world. Studies of human reader comprehension (Trabasso and Van Den Broek, 1985; Graesser et al., 1991, 1994) show that readers comprehend stories by tracking the relations between events. Specifically, reader comprehension relies on the tracking of at least four types of relations between events: (1) causal consequence, (2) goal hierarchies, (3) goal initiation, and (4) character intentions. The perceived coherence of a story is thus a function of the reader being able to comprehend how events correlate to each other causally or how they follow characters' pursuits of implicit goals. We hypothesize that a story generation system that makes decisions on how to continue a story based on tracking and reasoning about events will generate more coherent stories.

Unfortunately, stories don't always explicitly declare the causal consequences of events or the goals and intentions of characters. That is, sentences describing character actions or external events are rarely explicitly annotated with the characters' motivations and goals. Readers must infer the characters' goals, the relationship between their actions and those goals, and how their goals change as a result of the events in their world. The ability to use basic knowledge about goals and about world states falls within the study of *commonsense inference*. Initial work in this area was limited to modeling dimensions of the "naive psychology" of characters: motivations and emotional reactions (Rashkin et al., 2018). This was later extended to more attributes—ATOMIC (Sap et al., 2019) and ATOMIC$^{20}_{20}$ (Hwang et al., 2021) are event-centric commonsense knowledge bases that contain logical relationships between events and the mental states and attributes of their participants, represented as typed *if-then* relations. The former contains 9 such dimensions, and the latter 23. COMET$^{20}_{20}$ (Hwang et al., 2021), an extension of COMET (Bosselut et al., 2019), is a transformer-based generative model trained on triples from ATOMIC$^{20}_{20}$. Given a sentence, COMET$^{20}_{20}$ infers commonsense attributes about the characters that fall into three categories: (1) social interactions, (2) physical entities, and (3) effect of events inferred from the sentence. We hypothesize that a neural language generator informed about COMET-inferred event effects as well as character intentions and goals can generate more coherent narratives.

To this end, we introduce ***C**ommonsense inference **A**ugmented neural **S**tory**T**elling* (CAST), which infers the causal relations between events as well as the intents and motivations of characters in the story so far in order to generate story continuations that are more coherent to readers. CAST is a straightforward, cognitively inspired method to scaffold the generation of story text when sampling from a language model. By chaining sentence-level COMET inferences to track important implicit elements of the story over time, CAST is able to make more informed choices when sampling story continuations from a neural language model of choice (GPT-2 in our experiments). It can be used to produce both single-character and multiple-character stories. We hypothesize that stricter, more explicit constraints during generation should result in more coherent narratives than generating via sampling from a distribution alone, even if the distribution is fine-tuned. An overview of our method is presented in Figure 1.

To evaluate the efficacy of our proposed method, we conduct a series of human-participant experiments that measure perceptions of logical coherence of CAST against three strong neural language model story generators on three different story cor-

pora. Results indicate that the CAST method produces significantly more coherent, on-topic, enjoyable and fluent stories in both the single-character and two-character settings. This result holds even in a genre with a very different type of commonsense than that which COMET is trained on (fairy tales), indicating our method's generality.

2 Related Work

In addition to the work mentioned in the introduction, we provide a detailed background on story generation systems that emphasize commonsense reasoning and other related techniques. Guan et al. (2019) were the first to propose to incorporate a commonsense knowledge base into the story generation pipeline. Guan et al. (2020) improved upon this method by using the ATOMIC dataset to fine-tune GPT-2, and then fine-tuning a second time on the ROCStories corpus (Mostafazadeh et al., 2016). This system used multi-task learning during a second fine-tuning stage with an auxiliary objective to distinguish true and engineered false stories.

Similarly, Paul and Frank (2021) finetune GPT-2 on ROCStories to obey coherence rules generated by separately trained models. At inference-time, the story generation model is fed the first two and the last sentences of ROCStories test instances, making this an infilling rather than open-ended generation task. Brahman and Chaturvedi (2020) finetune GPT-2 to generate stories that follow a given emotional arc for a protagonist, using COMET to infer the protagonist's emotions as labels for their training dataset. They assume five emotions (anger, fear, joy, sadness, neutral) limited to two changes throughout the story, associated with the *xReact* and *oReact* inferences produced by COMET. We do not assume a fixed set of commonsense inference values, and we assume a character's state may change at each new sentence.

The C2PO system (Ammanabrolu et al., 2021) uses COMET to generate successor and predecessor events instead of a language model, performing a bi-directional search from a given start event and a given end event. C2PO assembles the narrative directly from the short, templated sentences produced by COMET. It also assumes the end of the story is known in advance. Like Brahman and Chaturvedi (2020), it focuses on only two dimensions of COMET and only works for a single character. Our work models interactions between multiple characters and takes advantage of a richer set of inferences that COMET and COMET$^{20}_{20}$ provide, better aligning with the four types of relations key to reader comprehension (Trabasso and Van Den Broek, 1985; Graesser et al., 1991, 1994).

Very recently, Lin et al. (2022) utilizes a BART-based commonsense inference model in conjunction with an event generation model to place event-related constraints on the story generation process. We acknowledge high-level similarity in approaches between our framework and this work, but we do not include this approach in our comparisons since it is concurrent.

Storytelling research focused on improving long-range cohesion is not limited to using commonsense resources—Goldfarb-Tarrant et al. (2020) perform high-level planning via plot outline generation using principles from Aristotle's *Poetics*, then use a language model in fill in details. They demonstrate strong performance on the WritingPrompts (Fan et al., 2018) dataset. While one of their models' purpose is to determine whether to reuse or introduce a new character, they do not explicitly model inter-character relationships or character attributes during generation. We compare to this baseline in our experiments.

3 The CAST Inference Method

We now introduce our neural storytelling framework, ***C**ommonsense inference **A**ugmented neural **S**tory**T**elling* (CAST), which scaffolds the conventional text generation process by imposing constraints on the sampling process at inference time.

The conventional setup for k-sentence story generation starts with a given first sentence s_1, referred to as the prompt, and generates $k-1$ subsequent sentences conditioned on it. CAST follows this convention, generating the ith sentence as follows:

1. We condition a fine-tuned language model on the story up to the current sentence $[s_1, \ldots, s_{i-1}]$, followed by a token signifying the main character of sentence i (§3.1).
2. We obtain a set of commonsense inferences for s_{i-1} (§3.2) and use them as constraints at the decoding stage of sampling a next-sentence candidate c from the language model (§3.3).
3. We obtain a set of commonsense inferences for candidate c, and match commonsense inference sets between s_{i-1} and c using a matching criteria, producing a score for c (§3.2).
4. If the score is above a specified threshold, c is selected to be s_i and is appended to the gener-

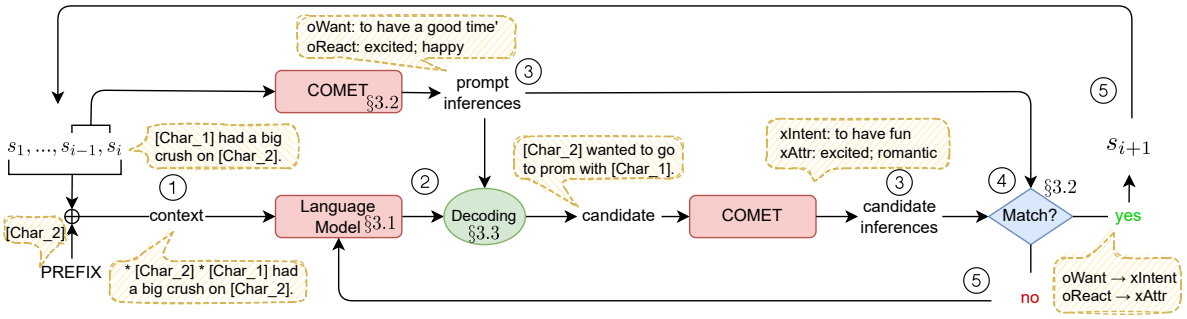

Figure 2: The overall procedure of generating two-character narratives with the CAST pipeline. 1. Condition the language model with story history and next character (§3.1). 2. A candidate continuation is generated. 3. Inferences on candidate continuation and story history are generated (§3.2). 4. Match commonsense inference sets. 5. If finding a match, the candidate is appended to story history; if not, repeat 2 through 4.

ation history. Otherwise, steps 2 through 4 are repeated until a viable candidate is found.

5. We repeat steps 1 through 4 until $k-1$ sentences have been generated.

An illustration of the pipeline is given in Figure 2. In practice, when generating two-character stories, we specify main characters in an alternating manner to promote turn-taking (more details in §3.1). CAST is not limited in application to the maximum story length seen during training, and can be used to generate stories of arbitrary length.

3.1 Language Model

We fine-tune GPT-2 on a story corpus to prime the model to elicit story-like generations (See Section 4.1 for details on the three story corpora we use for experiments). In order to directly compare to prior work (Guan et al., 2020), we use the small version of GPT-2, although our technique works with any neural language model.

We first pre-process the corpus to remove character names to improve generality and avoid gender bias. We replace them with character tags such as `[Char_1]`, `[Char_2]` or `[Char_3]`. This allows our generated stories to not be limited to turn-taking between characters of different genders, in contrast to prior work (Guan et al., 2020) who use gendered tags (`[MALE]`, `[FEMALE]`, `[NEUTRAL]`) to differentiate characters.

We perform a second fine-tuning step where we append a special prefix $*T*$ to sentence pairs. T is a character tag (`[Char_1]`, `[Char_2]`, etc.) representing the character that is the subject of the second sentence. Fine-tuning on this corpus allows us to specify main characters during Step 1 of the CAST inference process using the $*T*$ tag. This allows the model to generate a second sentence where T is the subject, but not necessarily the first word, of the sentence. For example, consider the sentence "`[Char_1]` has a big crush on `[Char_2]`". If the next sentence in the corpus has the entity represented by `[Char_2]` as the subject of the event , then we concatenate `*[Char_2]*` onto the first sentence during fine-tuning in order to cue the language model about turn-taking. We identify the subject entity in a sentence using a parser. We found in initial experiments that this allowed more flexibility and improved generation quality over the alternative (always requiring the main character T to be the first word of a sentence). More details are given in Appendix A.3.

To use the language model to generate a single-character story, we always set T to the same character (`[Char_1]`). In a two-character story setting, we adopt character turn-taking principle: in a two-character 5-sentence story, $T =$ `[Char_2]` for generating even numbered sentences and $T =$ `[Char_1]` for generating odd-numbered sentences.

3.2 Generating and Matching Commonsense Inferences

To produce commonsense inferences for each sentence, we use the COMET$^{20}_{20}$ model (Hwang et al., 2021) to infer a set of commonsense relations for each prompt sentence s_{i-1} and continuation sentence c. Table 1 has the list of the subset of inferences we use and their definitions.

Once we have inferred relation sets for both sentences, we look for specific patterns between the sets of ATOMIC relations. We identify eight relation pairs that are useful for creating coherent relations between story events described in adja-

Type	Definition
xWant	as a result, PersonX wants
xIntent	because PersonX wanted
xNeed	before this event, PersonX needed
xEffect	as a result, PersonX will
xAttr	PersonX is seen as
xReact	as a result, PersonX feels
oReact	as a result, PersonY or others feels
oWant	as a result, PersonY or others want
oEffect	as a result, PersonY or others will
CausesDesire	This event makes PersonX want
Desires	PersonX desires

Table 1: Definitions of the selected types of ATOMIC$^{20}_{20}$ relations that CAST uses. Those prefaced with x refer to the sentence's subject character and those with o to other characters.

# Characters	Prior Context	Continuation
Single	xWant xReact xEffect xReact CausesDesire	xIntent xReact xEffect xAttr Desires
Multiple	oReact oWant oEffect	xAttr xIntent xEffect

Table 2: Commonsense relation pairs that we identify as leading to meaningful sentence continuations. Table 1 provides attribute definitions.

cent sentences, five for single-character and three for two-character stories (Table 2) by analyzing semantic similarities between these relation pairs inferred by stories in ROCStories (Mostafazadeh et al., 2016). Details on the process for finding relation pairs can be found in Appendix A.4.

The relation types in the second column of Table 2 are interpreted as *postconditions* of a prior event because they are inferences of things that might have changed, such as an effect of the event, or a character is expected to form a new intention, or a character is expected to have a reaction. The relation types in the third column are interpreted as *preconditions* of any generated continuation because they are inferences about facts that needed to be established by prior events in the story, such as a character having an intention, a character having desires, or a character having a property.

CAST seeks to chain preconditions of the currently generated sentence with the post-conditions of the previous sentence as a means of checking whether readers will comprehend the continuation and perceive coherence. For example, suppose sentence s_{i-1} is "[Char_1] gives [Char_2] a burger". From this one can infer an oWant is "to thank", indicating that [Char_2] may want to thank [Char_1]. If [Char_2] is to be the subject of the subsequent sentence, a good candidate sentence would be one from which the xIntent "to thank" can be inferred, such as "[Char_2] said thanks to [Char_1]".

Once we have inferred relations for the previous sentence s_{i-1} and a current candidate c, we judge the coherence of c with the following procedure:

- The event in s_{i-1} affects the wants of a character, which manifests as an intention of the primary character in the subsequent sentence (xWant→xIntent in a single-character story; oWant→xIntent in a multi-character story).
- An effect of the event in s_{i-1} is something the primary character will do in the subsequent sentence (xEffect→xEffect in a single-character story; oEffect→xEffect in a multi-character story).
- A reaction to the event in s_{i-1} should match either some property, or the reaction, of the primary character in the subsequent sentence (xReact→xAttr/xReact in a single-character story and oReact→xAttr in a multi-character story).
- The character's desire in s_{i-1} should be consistent in the subsequent sentence (CausesDesire→Desires in a single-character story).

To filter out "unqualified" continuations generated by the language model, we match the inference types described in Table 2 and their arguments.

In practice, we find that simple string matching does not adequately capture when two inferred relations' arguments have slightly different phrasing (e.g., "to sleep" versus "sleeping"). We define a match as the semantic similarity between two inferences exceeding a certain threshold. To do this, we encode each relation argument into a fixed-length vector representation, and then compute the cosine similarity. We use Sentence-BERT (Reimers and Gurevych, 2019) for encoding, as it is designed for semantic similarity tasks and performs better than the traditional BERT on sentence similarity benchmarks. We use 80% semantic similarity as our threshold. In order to balance computation time and quality of the match, we require three of five and three out of three inference type pairs to match between a prompt and a candidate sentence when generating single-character and two-

character stories, respectively. Details of ablation studies on these hyperparameters can be found in Appendix A.5.

3.3 Increasing Sampling Success

CAST produces many candidates c—this step can be very expensive with respect to the average number of continuations needed to find a match (see Appendix A.5). In order to increase the probability of generating a continuation with a match, we use the commonsense inference set of prompt sentences as lexical constraints to control the decoding process (Peng and Sollami, 2022) when generating candidates. We first obtain the synonyms set A and the antonyms set $\bar{A}$ of each commonsense inference output by COMET using WordNet (Miller, 1995). For example, if COMET infers a want for a character of "go to beach", then $A =$ {"move to beach","go to beach"} and $\bar{A} =$ {"leave beach"}. We then expand the synonyms and antonyms sets by adding conjugated forms of verbs and the plural and singular forms of nouns (De Smedt and Daelemans, 2012). For example, {" buy dog"} is expanded to {" buy dogs", "buy a dog" , "buys a dog", "bought a dog", ...}. Finally, we modify the conditional probability $P_\theta(x_i \mid x_{<i})$ of the language model prior to sampling as follows:

$$P'_\theta(\vec{x_i} \mid x_{<i}) = P_\theta(\vec{x_i} \mid x_{<i}, A, \bar{A}) = \delta(\vec{x_i}, A, \bar{A}) \times P_\theta(\vec{x_i} \mid x_{<i})$$

where

$$\delta(\vec{x_{ij}}, A, \bar{A}) = \begin{cases} 1+\mu & \text{if } x_{ij} \in A, x_{ij} \notin \bar{A} \\ 1-\mu & \text{if } x_{ij} \notin A, x_{ij} \in \bar{A} \\ 1 & \text{otherwise} \end{cases}$$

and μ is a hyper-parameter to control the strength of the penalty. Using this altered distribution, we encourage productions of synonyms to COMET inferences and punish productions of antonyms. We only modify the probability of the *top-k* tokens in order to maintain the fluency of generated sentences. More details are in Appendix A.6.

4 Evaluation

4.1 Datasets

We conduct experiments on three diverse storytelling datasets:

- ROCStories (ROC) (Mostafazadeh et al., 2016): contains 98, 159 crowdsourced five-sentence stories involving common-sense scenarios.

Seed Prompt:
Alice invited Megan and the girls over for a clambake.
CAST:
The rain was so bad that Megan **couldn't play** in the rain. Alice got so **sad** and wanted to play in the rain with Megan. Megan got a **rain coat** to go outside. Alice finally **went outside** with Megan together.
GPT-ROC-RL:
Alice and Megan decided to ***stay inside*** for a week. Alice and Megan **bathroom was broken**. Alice and Megan got in so much **trouble**. Alice and Megan decided to *stay inside*.
Guan et al. (2020)
Megan tried to be friendly with each other. Alice each could feel their favorite animal tense in their hands. Megan caught up with them. Alice wrote down the activities on a topic they would enjoy.

Table 3: Story examples generated by CAST, CAST-RL, GPT-ROC and Guan et al. (2020). The story generated by CAST follows a single topic (**bolded**)—playing outside, and shows a good plot coherence. GPT-ROC-RL generates relatively more repetitive/boring but logically coherent narrative (in *italic*). Guan et al. (2020) suffers plot incoherence (underlined). More examples are given in Appendix C.

- Writing Prompts (WP) (Fan et al., 2018): ~300K user-generated stories along with their associated prompts from Reddit (`r/WritingPrompts/`). Average story length is 59.35 sentences.
- Fairy tales (FT) (Ammanabrolu et al., 2020a): 695 stories in the fairy tale genre scraped from story summaries on Wikipedia. Average length of stories is 24.80 sentences.

4.2 Baselines

We evaluate CAST against three strong baselines.

- Guan et al. (2020): fine-tunes GPT-2-Small on ATOMIC and ROCStories using a multi-objective training procedure. This baseline serves to demonstrate whether a neural language model can get everything it needs directly from a static commonsense dataset without inference and constraints. We retrain the model on the pre-processed version of the ROCStories corpus that does not contain gender tags (§3.1) as well as with the additional fine-tuning step for character-conditioned generation (subsection A.3), in order to be directly comparable to CAST in a two-character setting. Further training details can be found in Appendix A.2.
- Goldfarb-Tarrant et al. (2020): a plot-generation language model along with an ensemble of

Models	Data set	Num chars	Logical Sense			Single Topic			Enjoyable			Fluency		
			Win%	Lose%	Tie%	Win%	Lose%	Tie%	Win%	Lose%	Tie%	Win%	Lose%	Tie%
CAST vs Guan et al.	ROC	1	**92.0****	4.0	4.0	**86.0****	7.0	7.0	**87.0****	4.0	9.0	**87.0****	4.0	9.0
		2	**85.8****	6.6	7.5	**82.9****	8.6	8.6	**81.1****	12.3	6.6	**83.0****	9.4	7.5
CAST vs Goldfarb-Tarrant et al.	WP	1	**64.2***	32.1	3.8	**64.2****	28.3	7.5	**62.3****	26.4	11.3	**52.8**	34.0	13.2
CAST vs C2PO	FT	1	**81.5****	9.3	9.3	**63.6****	23.6	12.7	**81.8****	10.9	7.3	**85.5***	5.5	9.1

Table 4: Human-participant evaluation results for experiments 1 and 2, showing the percentage of participants who preferred the first system, second system, or thought the systems were equal. Each system is conditioned on the same test-set prompts. * indicates results are significant at $p < 0.05$ confidence level; ** at $p < 0.01$ using a Wilcoxan sign test on win-lose pairs. See results about majority votes and agreement in Table 10.

rescoring models on Writing Prompts dataset. The system trained BART (Lewis et al., 2020) to learn to generate plots on the given prompt and then transform them into a story. We compare CAST to Goldfarb-Tarrant et al. (2020)—one of the strongest story generators on Writing Prompts dataset—to show that CAST can be generalized to other datasets.

- C2PO (Ammanabrolu et al., 2021): uses COMET to generate successor and predecessor events for a single character, performing a bi-directional search from a given start event and an end event. It uses COMET to generate successor and predecessor events directly instead of constraining a more conventional language model as is the case with CAST. As such it is a strong baseline, especially considering it uses an extra piece of input—the story ending—that can influence perceptions of coherence. For fair comparison, we follow Ammanabrolu et al. (2021) to extract high-level plots from fairy tale stories and then use the first plot as prompt and the second plot as goal for guiding C2PO.

We use the provided checkpoints of the latter two models.[2] We thus only evaluate these systems on single-character stories, since C2PO is single-character story generator and Goldfarb-Tarrant et al. (2020) is not trained to generate stories with the number of characters chosen by humans.

4.3 Metrics

Given the well-established unreliability of automated metrics[3] for creative text generation, human-participant evaluation is generally held as the gold-standard evaluation technique (Celikyilmaz et al., 2020; Caglayan et al., 2020; van der Lee et al., 2021). Consequently, we also use human-participant evaluation. We provide human participants with a pair of stories from two systems, and ask them the following questions modified from Purdy et al. (2018):

- Which story better FOLLOWS A SINGLE TOPIC: for insight into perceptions of global coherence for the entire story.
- Which story's sentences MAKE MORE SENSE given sentences before and after them: to evaluate local causality and commonsense reasoning in the story.
- Which story is more ENJOYABLE: indicates story value and interestingness.
- Which story uses more FLUENT language: indicates story readability and grammaticality.

Similar questions have been used in evaluations of other story generation systems (cf. Tambwekar et al., 2019; Ammanabrolu et al., 2020b, 2021; Castricato et al., 2021). Each pairwise comparison is seen by at least 5 participants.

We conduct our studies using the Cloud Research crowdsourcing platform to interface with Amazon Mechanical Turk (Litman et al., 2017). Only those who pass a screening question are qualified for the study. Participants must also explain their preferences for each comparison with more than 50 characters of free text. We manually verify screening question responses to qualify participants and disregard data for those who fail the screening. All crowdsourcing studies we conducted were approved by our institution's Institutional Review Board (IRB). We recruited 86 participants from the

[2] https://github.com/PlusLabNLP/story-gen-BART; https://github.com/rajammanabrolu/C2PO

[3] Perplexity and BLEU (Papineni et al., 2002) scores are not applicable to evaluate CAST, because CAST is unconstrained neural language model story generator. It is not required to produce the same story with the gold story in the datsets. Self-BLEU (Zhu et al., 2018) measures frequency of words and bi-grams, which does not necessarily entail plot-level repetition; the same entities can make appearances in different events in different ways.

United States, paying $11.7 per hour on average. Only those with HIT approval rate above 90% and have over 1000 HITs approved were selected. Average inter-annotator agreement, measured by Fleiss' kappa (Fleiss, 1971), is > 0.2 (fair); a more detailed breakdown by experiment can be found in Table 10. Further details are provided in Appendix B.

We randomly select a subset of first-sentences from the test sets of each dataset—20 each of 1-character and 2-character prompts from ROCStories, 10 prompts from WP, and 10 from FT. We use these sentences to generate a story continuation of 4 sentences from each system.[4] We recruited 86 participants on a crowdsourcing platform. Each participant answered the four pairwise comparison questions (§4.3) on a randomly selected subset of 5 story pairs, comprised of one story from CAST and one from one of the baselines.

4.4 Results

The results are shown in Table 4 (top) where we detail the percentage of times human participants choose the story from one system over another for each dimension in the questionnaire. We indicate when results are significant at $p < 0.05$ and $p < 0.01$ confidence levels. Generally, participants strongly preferred stories generated by CAST to those generated by alternatives.

Compared with Guan et al. (2020), CAST is able to find a commonsense inference link to develop the stories on ROCStories prompts, which makes it much more coherent and stay on one single topic. Human participants state in their response that stories generated by CAST have better commonsense flow and make more sense. Stories generated by CAST is also more enjoyable and fluent because of its high coherence.[5]

Since COMET is trained on ROCStories, we also seek whether CAST works on other datasets. We compared CAST to Goldfarb-Tarrant et al. (2020) on Writing Prompts, which contain longer and more complicated stories. CAST with language model fine-tuned on Writing Prompts outperforms Goldfarb-Tarrant et al. (2020) in "Logical Sense", "Single Topic" and "Enjoyable" dimensions. On the topic of fluency, CAST is preferred but the result is not statistically significant when ties are considered. Human participants stated that they found stories generated by CAST is much easier to follow and they are built on a single topic. Because Goldfarb-Tarrant et al. (2020) applies BART (Lewis et al., 2020) to generate plots, which cannot ensure commonsense like CAST.

C2PO is also built on COMET to conduct a bi-directional search from a given start event and a given end event, which makes it as a strong baseline to compare. We follow Ammanabrolu et al. (2021) to extract high-level plots from fairy tale stories as prompts and goals[6] for evaluation. CAST outperforms C2PO on all dimensions, because we apply a harder commonsense constraints on continuation generation than C2PO, which produce a more coherent and on-topic story. We anecdotally observe that CAST generates more diverse stories than C2PO because of templated and limited range of COMET, which we only use for filtering whereas C2PO uses it for sentence generation.

We conclude that CAST is able to produce a much more coherent, on-topic, enjoyable and fluent story than strong baselines. It also has the advantage over Goldfarb-Tarrant et al. (2020) and C2PO for choosing characters in the story continuations, which makes CAST able to produce single- or two-character stories.

5 Conclusions

Neural language models generate content based on the likelihood of tokens given a historical context. Human readers, on the other hand, use complex inferential processes to connect the relationships between events. This mismatch between generative models and reader comprehension is one of the reasons why stories generated by neural language models lose coherence over time.

Our CAST system is a straightforward approach to enforce the constraint that a language model only generate continuations that cognitive psychology tells us will be more comprehensible. The CAST method provides hard constraints to neural language model generation that results in greater story coherence, a result that holds in multiple storytelling domains. We find that perceived story enjoyability and fluency are tied to making logical sense, tracking character goals, and staying on topic; our system excels in all four of these areas.

[4] We seed prompts and goal sentences for C2PO. For 2-character stories, we use the interleaving story generation method described in §3.1.

[5] We observe that these four dimensions are highly, positively correlated using Spearman's Rank Order Correlation (See Appendix B.3).

[6] We only provide prompts to CAST.

6 Acknowledgements

This work was done while SW was at the Georgia Institute of Technology.

7 Limitations

The primary data source of our paper is the ROCStories dataset. ROCStories consists of many event-centric narratives which, while often used in story generation research, is still not representative of complex, realistic narratives. This may give COMET^{20}_{20} (Hwang et al., 2021) an advantage in making inferences that are used for filtering.

COMET^{20}_{20} requires a clearly identifiable actor in each sentence in order to make commonsense inferences for that actors. Thus our language model—by virtue of fine-tuning on ROCStories—produces sentences (events) that have an identifiable character peforming an action. Stories can have more complex expository text. Narratologists—those that study narratives—often distinguish between *events*—text that implies a change to the world and thus drive the story forward—and *exposition*—text that describes elements of the story world without changing it.

The performance of CAST is tied to the inference abilities of COMET^{20}_{20}. As such, the types of errors that COMET is prone to are also the types of errors that our system is prone to. We invite readers to review the discussion in Hwang et al. (2021) for more detailed analysis of commonsense inference errors. As more advanced commonsense inference models develop, CAST-like approaches will benefit from the improved state-of-the-art. CAST can easily switch to new generative language models or new commonsense inference model.

Restricted by the filter—COMET^{20}_{20}—our system works mostly for narratives with event-centric commonsense knowledge. Even though we processed the datasets (Appendix A.1) to decrease the gender biases, there is no guarantee to entirely eliminate these biases.

CAST produces stories by chaining sentence-level COMET inferences to track important implicit elements of the story between adjacent sentences. We make a Markovian assumption by only comparing the currently generated event to the most recent event. Stories are arguably non-Markovian and can have complex, interleaving chains of inference; despite the assumption, we find in practice that it enforces global coherence quite successfully (see Single Topic metric in Table 4). Future work may relax this constraint by keeping track of wants/needs/etc from previous sentences against which to match. One would need to solve the problem of deciding when wants/needs/etc should expire because they are no longer applicable.

References

Prithviraj Ammanabrolu, Wesley Cheung, William Broniec, and Mark O Riedl. 2021. Automated storytelling via causal, commonsense plot ordering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 5859–5867.

Prithviraj Ammanabrolu, Wesley Cheung, Dan Tu, William Broniec, and Mark Riedl. 2020a. Bringing stories alive: Generating interactive fiction worlds. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, volume 16, pages 3–9.

Prithviraj Ammanabrolu, Ethan Tien, Wesley Cheung, Zhaochen Luo, William Ma, Lara J Martin, and Mark O Riedl. 2020b. Story realization: Expanding plot events into sentences. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7375–7382.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Faeze Brahman and Snigdha Chaturvedi. 2020. Modeling protagonist emotions for emotion-aware storytelling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5277–5294, Online. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Ozan Caglayan, Pranava Madhyastha, and Lucia Specia. 2020. Curious case of language generation evaluation metrics: A cautionary tale. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2322–2328, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Louis Castricato, Spencer Frazier, Jonathan Balloch, and Mark O. Riedl. 2021. Tell me a story like i'm five: Story generation via question answering. In *Proceedings of the 3rd Workshop on Narrative Understanding*, Virtual. Association for Computational Linguistics.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of text generation: A survey. *arXiv preprint arXiv:2006.14799*.

Elizabeth Clark, Yangfeng Ji, and Noah A. Smith. 2018. Neural text generation in stories using entity representations as context. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2250–2260, New Orleans, Louisiana. Association for Computational Linguistics.

Tom De Smedt and Walter Daelemans. 2012. Pattern for python. *The Journal of Machine Learning Research*, 13(1):2063–2067.

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 (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

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*, pages 2650–2660, Florence, Italy. Association for Computational Linguistics.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Pablo Gervás. 2013. Propp's morphology of the folk tale as a grammar for generation. In *2013 Workshop on Computational Models of Narrative*.

Pablo Gervás. 2014. Composing narrative discourse for stories of many characters: a case study over a chess game. *Literary and Linguistic Computing*, 29(4):511–531.

Seraphina Goldfarb-Tarrant, Tuhin Chakrabarty, Ralph Weischedel, and Nanyun Peng. 2020. Content planning for neural story generation with aristotelian rescoring. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4319–4338, Online. Association for Computational Linguistics.

Art Graesser, Kathy L. Lang, and Richard M. Roberts. 1991. Question answering in the context of stories. *Journal of Experimental Psychology: General*, 120(3):254–277.

Art Graesser, Murray Singer, and Tom Trabasso. 1994. Constructing inferences during narrative text comprehension. *Psychological Review*, 101(3):371–395.

Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. 2020. A Knowledge-Enhanced Pretraining Model for Commonsense Story Generation. *Transactions of the Association for Computational Linguistics*, 8:93–108.

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

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 (ICLR)*.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. COMET-ATOMIC 2020: On symbolic and neural commonsense knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6384–6392.

Ahmed Khalifa, Gabriella AB Barros, and Julian Togelius. 2017. DeepTingle. In *Proceedings of the 8th International Conference on Computational Creativity*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Li Lin, Yixin Cao, Lifu Huang, Shuang Li, Xuming Hu, Lijie Wen, and Jianmin Wang. 2022. Inferring commonsense explanations as prompts for future event generation. *arXiv preprint arXiv:2201.07099*.

Leib Litman, Jonathan Robinson, and Tzvi Abberbock. 2017. Turkprime. com: A versatile crowdsourcing data acquisition platform for the behavioral sciences. *Behavior research methods*, 49(2):433–442.

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 AAAI Conference on Artificial Intelligence*, volume 32, pages 868–875.

George A Miller. 1995. WordNet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

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*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Jessica Ouyang and Kathleen McKeown. 2015. Modeling reportable events as turning points in narrative. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2149–2158, Lisbon, Portugal. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Debjit Paul and Anette Frank. 2021. COINS: Dynamically generating COntextualized inference rules for narrative story completion. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5086–5099, Online. Association for Computational Linguistics.

Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. 2018. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, New Orleans, Louisiana. Association for Computational Linguistics.

Xiangyu Peng and Michael Sollami. 2022. XFBoost: Improving text generation with controllable decoders. *arXiv preprint arXiv:2202.08124*.

Christopher Purdy, Xinyu Wang, Larry He, and Mark Riedl. 2018. Predicting generated story quality with quantitative measures. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, volume 14, pages 95–101.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Hannah Rashkin, Antoine Bosselut, Maarten Sap, Kevin Knight, and Yejin Choi. 2018. Modeling naive psychology of characters in simple commonsense stories. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2289–2299, Melbourne, Australia. Association for Computational Linguistics.

Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. 2020. PlotMachines: Outline-conditioned generation with dynamic plot state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4274–4295, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*

and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Mark O. Riedl. 2016. Computational narrative intelligence: A human-centered goal for artificial intelligence. In *Proceedings of the CHI 2016 Workshop on Human Centered Machine Learning*.

Mark O. Riedl and R. Michael Young. 2010. Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research*, 39:217–268.

Melissa Roemmele. 2016. Writing stories with help from recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, pages 4311–4312.

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 AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J Martin, Animesh Mehta, Brent Harrison, and Mark O Riedl. 2019. Controllable neural story plot generation via reward shaping. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 28, pages 5982–5988.

Tom Trabasso and Paul Van Den Broek. 1985. Causal thinking and the representation of narrative events. *Journal of memory and language*, 24(5):612–630.

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, and Emiel Krahmer. 2021. Human evaluation of automatically generated text: Current trends and best practice guidelines. *Computer Speech & Language*, 67:101151.

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 AAAI Conference on Artificial Intelligence*, volume 33, pages 7378–7385.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, volume 41, pages 1097–1100.

A Implementation Details

A.1 Data

We use the preprocessed ROCStories corpus of 5-sentence stories (588,966 stories) and joint ATOMIC and ConceptNet dataset converted to template sentences (1,174,267 train/66,856 dev/73,083 test), both provided by Guan et al.[7] We shuffle and split the ROCStories dataset into 80% (78,528) train and 20% (19,633) test sets.

Following Guan et al. (2020), character names in the ROCStories corpus are replaced with `[MALE]` or `[FEMALE]` tags. In order to remove gender bias, we replace `[MALE]`, `[FEMALE]`, or `[NEUTRAL]` tags with `[Char_1]`, `[Char_2]`, or `[Char_3]` tags. This prevents skewed predictions due to the presence of certain names in a small dataset such as ROCStories and also allows us to focus on 2-character stories without having to perform NER on generated sentences to remove extraneously generated names outside of the two main characters. It also allows a direct comparison to prior work. After a story is generated, we replace the character tags with user-inputted names, assuming the subject and object of the first sentence are the subsequently-generated tags.

A.2 Models

Following Guan et al., we use the small version of GPT-2 with 124M parameters as the base for our fine-tuned models. When fine-tuning GPT-2 on either ROCStories and the commonsense knowledge resources (done separately), we train with a learning rate of 0.00005, and using the Adam optimizer with gradient clipping at a max norm of 1. CAST and Guan et al. (2020) were trained on single GeForce RTX 2080 GPUs in Pytorch using the Huggingface Transformers library.[8] We replicate the multi-task baseline of Guan et al. in Tensorflow using their provided code.[9] We train with early stopping on the dev set (80% train,10% dev,10% test split) loss with a patience of 10 epochs. Both models converge within 1-2 epochs. All other training details are kept the same. We use top-p sampling (Holtzman et al., 2019) with a value of 0.9, a temperature of 1, and a max length of 20 tokens per sentence to sample from CAST and Guan et al. (2020).

For Goldfarb-Tarrant et al. (2020), we use BART model using the code and parameters published on the paper's public repository [10].

We replicate the C2PO model by Ammanabrolu et al. (2021) using the code published on the paper's public repository [11]. All the encoder and model checkpoints are provided by the author.

A.3 Character Conditioned Generation

To enforce the telling of a two-character narrative in an interleaving fashion wherein characters take turns being the subject of each sentence. We fine-tune the language model by formulating the input as $*\mathrm{T}*\ [s_1, \ldots, s_{i-1}]$, where T is the tag denoting the character who is to take a turn, which is determined by Part-Of-Speech Tagger[12]. For example, for the story, "`[Char_1]` was upset with `[Char_2]`. Because of this, `[Char_2]` apologized.", the prompt is formulated as "* `[Char_2]` * `[Char_1]` was upset with `[Char_2]`." The language model is fine-tuned by back-propagating the loss calculated on the sentence "Because of this, `[Char_2]` apologized." At test-time, we generate second sentence candidates until one contains a reference to `[Char_2]`.

A.4 Commonsense Matching Criteria

We randomly selected 500 stories from ROCStories(Mostafazadeh et al., 2016). We then use COMET to produce commonsense inference sets for these stories of all the 34 relations in ATOMIC with beam size of 10. Hence, for all the 500×5 sentences, we obtain 10 commonsense inference for each type (34 types). For each sentence, we consider the commonsense relation sets of current sentence and its next sentence as a pair. So we have 32×31 pairs for each pair of the adjacent sentences. Then we adopt Sentence-BERT (Reimers and Gurevych, 2019) to encode all these inference and calculate the max cosine similarity of each commonsense inference pair for each adjacence sentence pair. Inference pairs with over 80% semantic similarities are used as hard constraints via a form of chaining that allows us to filter a set of potential sentence generations to find one that adequately matches the expected inferences.

[7] https://cloud.tsinghua.edu.cn/d/670f7787b6554f308226/
[8] https://huggingface.co/transformers/
[9] https://github.com/thu-coai/CommonsenseStoryGen

Semantic Similarity	# of Sentence Candidates ↓	Success Rate ↑	Self ↓ BLEU-2	Self ↓ BLEU-3
0.8	11.25	98.75%	.1718	.0892
0.85	17.25	97.25%	.1978	.1140
0.9	18.05	91.25%	.2011	.1216

Table 5: Ablation study result for semantic similarity. We run CAST without controlling decoding to generate 20 2-char stories in 5 seeds. *# of sentence candidates* denotes the average number of sentences candidates generated before finding a matching inference type pair. *Success rate* is the percentage of finding a match within the 50-candidate limit. A lower Self-BLEU score implies more diversity of the document (Zhu et al., 2018) (see §4.3).

Seed Prompt: Bob was in love with boyfriend Alice.
Semantic Similarity = 0.8: Alice was in love with Bob since grade school. One day, Bob decided she would get married with Alice. Alice proposed to Bob. Bob and Alice got married.
Semantic Similarity = 0.9: Alice wanted to get **married**. Bob was asked to **marry** in the article. Alice prepared to get **married**. Bob proposed..
Seed Prompt: Bob 's cousin Alice wanted the kids to play outside in the summer.
Semantic Similarity = 0.8: But the kids had a crush on Alice and wanted to play in the yard. Bob had a good plan. Alice went to the store to get them new toys. Bob brought them to the car and went home with the new toys.
Semantic Similarity = 0.9: So Alice took the kids outside. Bob was surprised that Bob cousin was back. Alice loved playing with the kids. Bob was not the only one to play, and the two played happily.

Table 6: Story examples generated by CAST with different semantic similarity thresholds. Stories generated at 80% similarity maintains more diversity.

A.5 Ablation Study of Commonsense Inferences Matching

We use 80% semantic similarity as our lower-bound. Empirically, we find this value best considers the inferences listed in Section 3.2 as matches, but excludes less-related inferences. Table 6 shows how the threshold affects success rate—the percentage of queries that find a match within 50 generated candidates—and the diversity of results as measured by self-BLEU score (described in §4.3). Each system was conditioned on the same 20 2-character prompts from ROCStories with 5 different random seeds, requiring two of three inference

[10] https://github.com/PlusLabNLP/story-gen-BART
[11] https://github.com/rajammanabrolu/C2PO
[12] https://nlp.stanford.edu/software/tagger.shtml

# of Matching	# of Sentence Candidates ↓	Success Rate ↑	Self ↓ BLEU-2	Self ↓ BLEU-3
3	9.78	98.67%	.1559	.0860
4	18.95	86.83%	.1832	.1282
5	56.39	40.17%	.1870	.1356

Table 7: Ablation study result for required matching inference type pairs in single-character stories. We run CAST without controlling decoding to generate 30 single-character stories in 5 seeds. *# of sentence candidates* denotes the average number of sentences candidates generated before finding a matching inference type pair. *Success rate* is the percentage of finding a match within the 50-candidate limit. The number of sentence candidates could Failure to find a match within the candidates limit (50) will relax the matching constraints to one pair. Hence, the average number of sentences candidates might be over the candidate limit. A lower Self-BLEU score implies more diversity of the document (Zhu et al., 2018) (see §4.3).

Seed Prompt: Bob enjoyed long walks on the beach.
# of Matching = 3: Bob was always healthy and energetic. Bob enjoyed the sun and the heat. One day, Bob decided to take a walk in the beach. Bob had fun at the beach for the whole day.
# of Matching = 5: One day, Bob decided to go for a long walk on the beach. Bob loved the sun so much, Bob always happy. Bob enjoyed the sun when Bob walked on the beach. Bob liked Bob walk on the beach.
Seed Prompt: Bob had just learned how to ride a bike.
# of Matching = 3: Bob went to the store to buy a new bike. After buying a new bike, Bob went to ride it. Bob rode it to the park. Bob loved his new bike.
# of Matching = 5: Bob mom took Bob on a bike ride. Bob went on the bike for hours. Finally Bob was back on Bob bike. Bob loved riding it.

Table 8: Story examples generated by CAST with different semantic similarity thresholds. Stories generated at 80% similarity maintains more diversity.

type pairs to match to qualify as a match. Failure to find a match within the candidates limit (50) will relax the matching constraints to two pairs. Hence, the average number of sentences candidates might be over the candidate limit. As observed in 5 , increasing the semantic similarity threshold decreases the success rate in obtaining a matching candidate within the sentence limit, and it results in more repetitive sentences (see Table 6).

In order to balance computation time and quality of the match, we only require three of five inference type pairs to match between a seed and a candidate sentence when generating single-character stories. When requiring five matches when generating single-char story, CAST only finds a "qualified" sentence 40% of the time within 50 attempts (see Table 5 (bottom), computed at 0.8 semantic similarity). In practice (see examples in Table 8), we find requiring three pairs results in higher quality sentences than if we only require one or two out of three pairs to match, but is significantly more efficient than four or five out of five.

A.6 Decoding Process Ablation Study

Decoding Matching	# of Sentence Candidates ↓	Success Rate ↑	Self ↓ BLEU-2	Self ↓ BLEU-3
True	3.21	99.50%	**.1709**	.0913
False	11.25	98.75%	.1718	**.0892**

Table 9: Ablation study result for required controlling decoding stage. We run CAST with or without controlling decoding to generate 20 multiple-character stories in 5 seeds. *# of sentence candidates* denotes the average number of sentences candidates generated before finding a matching inference type pair. *Success rate* is the percentage of finding a match within the 50-candidate limit. A lower Self-BLEU score implies more diversity of the document (Zhu et al., 2018).

In order to increase the probability of finding a match in Section 3.2, inspired by Peng and Sollami (2022), we use commonsense inferences of prompt sentences as lexical constraints to control the generation decoding process. We run a ablation test to validate this component of CAST. Table 9 shows that after applying commonsense inferences of prompt sentences as lexical constraints to control the generation decoding process, CAST successfully find a match in the average of 3 candidates. At the same time, self-BLEU score did not show any statistical difference. Hence, we adopt decoding technique in CAST.

A.7 CAST

When producing commonsense inferences from COMET, we use "beam-5" setting to generate 5 inferences for each inference type, which results in a higher percent of matched inferences in our preliminary experiments. We also qualitatively observe that matching on a larger set of inferences (as shown in the demo[13]) more often results in at least one or a few high-quality inferences, due to COMET having some error.

As mentioned in the body of the text, we use a semantic similarity threshold of 80% and require 3 of 5 inferences to match when generating single-character stories. Runtime is feasible due to matching on three out of five inference filters and using the 5-beam COMET output. However, in some rare cases, no matching next-sentence candidate can be found. If no qualified sentence is found after 50 generated candidates, in order to avoid potentially infinite search we loosen the filtering strength to match only one pair of inferences. We also report the majority vote of experiments in Table 10.

[13] https://mosaickg.apps.allenai.org/comet_atomic

Models	Data set	Num chars	Logical Sense Win%	Lose%	Tie%	Single Topic Win%	Lose%	Tie%	Enjoyable Win%	Lose%	Tie%	Fluency Win%	Lose%	Tie%	Num story
CAST vs Guan et al.	ROC	1	**95****	0	5	**90****	5	5	**95****	5	5	**95****	5	5	20
		2	**90****†	5	5	**90****	5	5	**95****†	5	0	**95****†	5	0	20
CAST vs Goldfarb-Tarrant et al.	WP	1	**60**‡	30	10	**70**†	30	0	**70**‡	30	0	**60**†	40	0	10
CAST vs C2PO	FT	1	**90****†	10	0	**70**	20	10	**100****	0	0	**100***	0	0	10
GPT-ROC-RL vs Guan et al.	ROC	1	**65****†	15	20	**55**†	35	10	**50**†	40	10	**50**	30	20	20
		2	**60**†	30	10	**50**	40	10	**65**	30	5	**70***	15	15	20
GPT-ROC-RL vs CAST	ROC	1	15	**80****†	5	10	**80****†	10	20	**75****†	5	15	**65****	20	20
		2	5	**85****†	10	5	**80****	15	100	**85****†	5	0	**95****†	5	20

Table 10: Human-participant evaluation results for experiments 1 and 2, showing the percentage of participants who preferred the first system, second system, or thought the systems were equal. Each system is conditioned on the same test-set prompts. * indicates results are significant at $p < 0.05$ confidence level; ** at $p < 0.01$ using a Wilcoxan sign test on win-lose pairs. † indicates $\kappa > 0.2$ or fair agreement. ‡ indicates $\kappa > 0.4$ or moderate agreement.

B Evaluation

B.1 Evaluated Stories Generation

In order to compare with Guan et al. (2020), we randomly select a subset of first sentences of ROC-Stories as prompts to seed CAST and Guan et al. (2020), generating 5-sentence stories from each model. We considered two cases—(1) single-character and (2) two-character stories. In order to generate two-character stories, we seed the story history and continuation's subject to GPT-2. More details can be found in Appendix A.3. We use a subset of prompts given by Writing Prompts to seed our system and Goldfarb-Tarrant et al. (2020). We also keep 5 sentences to evaluate the models. Since C2PO is controllable story generation model trained on fairy tale stories, we seed the first sentence in the fairy tale stories as prompt and the 5th sentence in the story as goal to C2PO for generating stories. For CAST model, we only seed the first sentences of the fairy tale stories. Examples can be found in Appendix C.

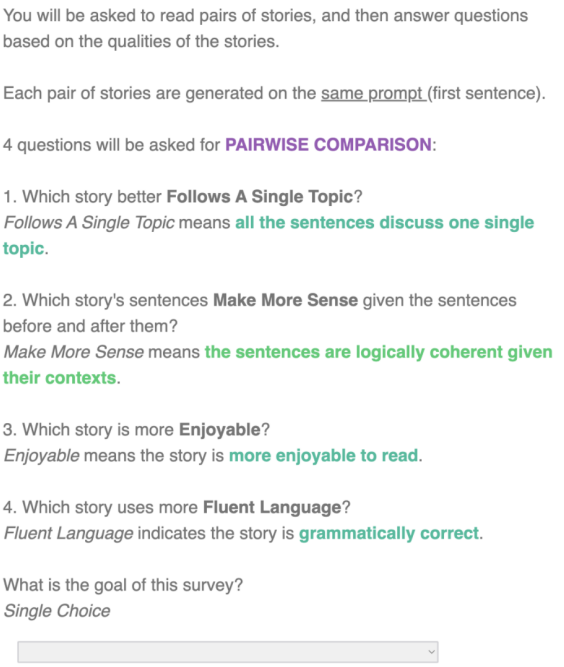

Figure 3: Instructions given to human study participants, along with a question to validate they have read them.

Table 11.

B.2 Human Study Setup

We show human participants instructions (Fig. 3) and then they are required to pass screen questions (Fig. 4). They then answer which story best met the criteria, as shown in Fig. 5.

B.3 Correlation between answers

We compute the Spearman rank correlations between the workers' different answers for the story pairs they are responsible for rating. We ignore workers who did not complete all of the questions in our computations. Our results are displayed in

Study	**1 & 2**	**1 & 3**	**1 & 4**	**2 & 3**	**2 & 4**	**3 & 4**
CAST vs. C2PO	0.39*	0.34	0.34	0.47*	0.77*	0.39*
CAST vs. Goldfarb-Tarrant et al.(2020)	0.58*	0.44*	0.38*	0.56*	0.35	0.47*
CAST-ROC-1Char vs. Guan et al.(2020)	0.51*	0.28*	0.43*	0.67*	0.65*	0.67*
CAST-ROC-2Char vs. Guan et al.(2020)	0.53*	0.49*	0.39*	0.91*	0.79*	0.79*
CAST-RL-1Char vs. Guan et al.(2020)	0.78*	0.65*	0.47*	0.67*	0.54*	0.45*
CAST-RL-2Char vs. Guan et al.(2020)	0.51*	0.45*	0.44*	0.65*	0.65*	0.55
CAST-RL-1Char vs. CAST-ROC-1Char	0.83*	0.28*	0.55*	0.32*	0.64*	0.31*
CAST-RL-2Char vs. CAST-ROC-2Char	0.14	-0.05	0.25	0.27	0.39*	0.30*

Table 11: Spearman correlation results from all our human subject studies. The four numbers 1, 2, 3, and 4 corresponds to "Follow a single topic", "Logical Sense", "Enjoyable", and "Fluency" metrics respectively. Spearman correlations with $p \leq 0.01$ are marked with an *.

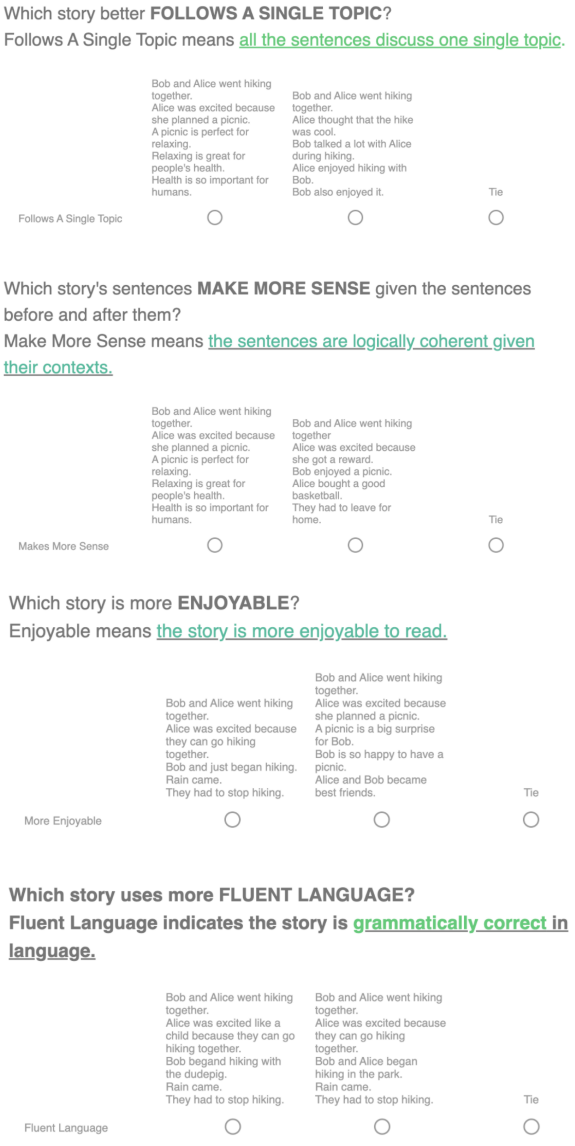

Figure 4: Screening questions used to qualify participants for the main study. Correct answers for these questions are [2,1,any,2], where 1 indicates the first story is the correct answer, 2 indicates the second one is the correct answer and "any" indicates no correct answers, any answer can pass the screen question.

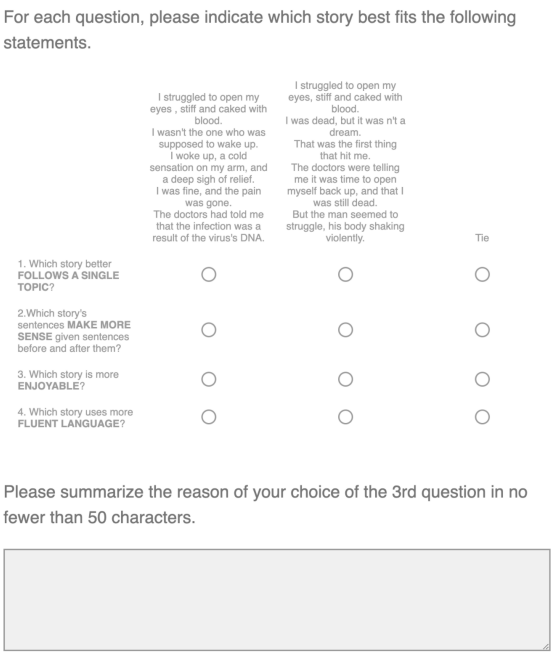

Figure 5: User interface for the main human evaluation task. Each participant completes 5 such tasks in a single HIT.

C Additional Story Outputs

Additional 5-sentence story examples are as follows,

Seed Prompt:
Bob and Alice went hiking together.

CAST:
Alice was excited because she planned a picnic.
Bob was excited because he was staying up late at night.
Alice felt much better about her camping trip.
Bob spent the night playing in the woods.

CAST-RL:
Alice decided to take a group photo.
Bob shared the picture with Alice.
Alice came over and smiled at Bob.
Bob was so happy that he took the picture.

Guan et al. (2020):
Alice thought that the hike was cool.
Bob was pretty cool.
Alice reminded him of the great fall, especially in the forest.
Bob was cute, but Alice was lost.

Seed Prompt:
Bob asked Alice to come over for dinner.

CAST:
Alice wanted to go to a party.
Bob agreed to go with her.
Alice and Bob went to the concert.
Bob bought a bottle of wine, and they enjoyed their time.

CAST-RL:
Alice said yes, so he brought her food.
Bob had dinner with Alice that night.
Alice said yes to the dinner.
Bob was happy that he invited her to dinner with him.

Guan et al. (2020):
Alice drove and came over Bob 's house in time.
Bob decided to let Alice choose the foods.
Alice ate some dog food and then she met her husband.
Bob offered Alice a side of anything he had for dinner.

Seed Prompt:
Bob took Alice fishing in the summer.

CAST:
Alice waited at the lake for hours.
Bob got another boat and brought it back home.
Alice and Bob fished the next day.
Bob enjoyed the fishing and had a great time.

CAST-RL:
Alice caught ten fish.
Bob was pleased that he caught ten fish.
Alice was very pleased to have a good fish.
Bob is glad that he caught ten fish.

Guan et al. (2020):
Alice got seasick.
Bob asked Alice to help him.
Alice went to Bob 's house to tow Bob 's boat.
Bob got a surfboard for a big fishing trip.

Seed Prompt:
Bob was engaged to Alice.

CAST:
Alice wanted a diamond ring.
Bob thought that ring was the perfect diamond.
Alice loved the diamond ring and decided to buy it.
Bob went to the store and bought the ring.

CAST-RL:
Alice was going on a date with Bob.
Bob agreed to the date and married her.
Alice now has two kids and needs the baby.
Bob is very happy he finally realized she was unhappy.

Guan et al. (2020):
Alice just got a new husband.
Bob wondered what to expect.
Alice bought the nice wedding ring from a daycare.
Bob wore it to the wedding.

Seed Prompt:
Bob took Alice to school today.

CAST:
Alice was a freshman.
Bob took her to the library.
Alice was excited to take the long bus ride to the library.
Bob continued the long walk through the library.

CAST-RL:
Alice had been waiting for hours at school.
Bob was relieved to see her take the bus to school.
Alice was excited to see the bus arrive.
Bob was proud of her for taking the bus to school.

Guan et al. (2020):
Alice was nervous because Bob didn't know how to tap dance.
Bob was excited for his first attempt at tap dancing.
Alice was impressed and wanted to learn.
Bob started his first dance at the dance school.

Additional 10-sentence story examples are as follows:

Seed Prompt:
Bob went on her very first date with Alice.

CAST:
Alice was very excited about having a date with Bob.
Bob was excited and eager.
Alice informed Bob she was very excited for him.
Bob was very appreciative.
Alice said it was a perfect time.
Bob and Alice had a wonderful first date.
Alice was surprised she was invited to his birthday.
Bob was thrilled.
Alice was happy to have a new boyfriend.

Guan et al. (2020):
Alice and Bob didn't realize that Bob loved movies.
Bob's friend, Alice, saw the movie with Alice.
Alice and Bob were good friends.
Bob's friend, Alice, screamed for Alice.
Alice was their eyes met her eyes.
Bob and Alice's grandparents were never match.
Alice because they had ever though, they thought the end the words said, she passed with Alice.
Bob and they agreed that they went out the eyes.
Alice felt they got puppies and they do the thought she was their eyes Alice had.

Seed Prompt:
Bob broke up with Alice.

CAST:
Alice was very hurt and very angry.
Bob knew he would have to live with his pain.
Alice said she could give him a relief.
Bob felt much better after that.
Alice felt even better in peace.
Bob is now happy with himself.
Alice felt much better.
Bob will make himself a happy man.
Alice feels much better after having peace.

Guan et al. (2020):
Alice didn't want to get it.
Bob thought that he was too thin.
Alice said he could give back inside a horse.
Bob could bring along for a ride and to return to the river.
Alice had to drive for three days and she realized she was cheating.
Bob and Alice got into a fight.
Alice ran away with Alice and grounded.
Bob and sent Alice.
Alice and a girl kicked each other.

Seed Prompt:
Bob went to see Alice today.

CAST:
Alice told him about the movie she loved to watch.
Bob grabbed a movie poster from the library.
Alice shared her own reaction to the movie.
Bob watched the movie that night.
Alice felt very satisfied after the movie was over.
Bob went home happy that he watched the movie.
Alice has been a huge fan of the movie ever since.
Bob looks forward to a nice vacation at the movies.
Alice is glad that she shared her own comments to the movie.

Guan et al. (2020):
Alice was nice and gregarious.
Bob had a great time.
Alice had some great food and wine.
Bob couldn't wait to see Alice again.
Alice let him take a bath.

Bob felt smarter and had a good time again.
Alice also got his great seeing Alice tattoo.
Bob had a good time.
Alice tattooy from his favorite j tomorrow too.

Seed Prompt:
Bob went over to play at Alice's house.

CAST:
Alice liked to watch movies.
Bob was always there watching movies.
Alice saw a movie she wanted to see.
Bob was very glad to buy that movie.
Alice was so happy with the movie he bought.
Bob was excited to be watching the movie with Alice.
Alice could not wait to see the movie with Bob.
Bob and Alice had so many fun.
Alice laughed and thanked Bob for the great movie.

Guan et al. (2020):
Alice was a baby dolls.
Bob told Alice he could save a doll.
Alice went to the video game section to change the game.
Bob was amazed to see who did the laundry to the last minute.
Alice waved and proposed the doll girls.
Bob wished her that she was getting a doll.
Alice helped him with the $40 and she gave her $100, but by buttering buttering a doll.
Bob's doll broke.
Alice had an ruined doll and they would buy her.
