Returning to the Start: Generating Narratives with Related Endpoints

Anneliese Brei Chao Zhao Snigdha Chaturvedi
Department of Computer Science, UNC Chapel Hill
{abrei, zhaochao, snigdha}@cs.unc.edu

Abstract

Human writers often *bookend* their writing with ending sentences that relate back to the beginning sentences in order to compose a satisfying narrative that "closes the loop." Motivated by this observation, we propose RENARGEN, a controllable story-generation paradigm that generates narratives by ensuring the first and last sentences are related and then infilling the middle sentences. Our contributions include an initial exploration of how various methods of bookending from Narratology affect language modeling for stories. Automatic and human evaluations indicate RENARGEN produces better stories with more narrative closure than current autoregressive models.

1 Introduction

Narrative closure is an important feature of satisfying narratives. Carroll (2007) defines narrative closure as "the phenomenological feeling of finality that is generated when all the questions saliently posed by the narrative are answered." Human writers often achieve closure through *bookending* (Adamo, 1995) (a.k.a circular construction or ring composition) whose minimum criteria is for the ending to relate back to the beginning (Novakovich, 2008; Katz, 2023).

Automatic story generation has advanced significantly recently (Chaturvedi et al., 2016, 2017; Peng et al., 2017; Fan et al., 2018; Yao et al., 2019; Fan et al., 2019; Brahman and Chaturvedi, 2020; Brahman et al., 2020; Freiknecht and Effelsberg, 2020; Castricato et al., 2021; Chowdhury et al., 2021; Vijjini et al., 2022; Yang et al., 2022; Huang et al., 2023). However, these approaches still struggle to generate satisfying and coherent stories with closure (Alabdulkarim et al., 2021; Piper et al., 2021). To address this challenge, we propose **Re**lated **E**ndpoint **Nar**rative **Gen**erator (RENARGEN)[1] to generate closed narratives via bookending with related first and last sentences.

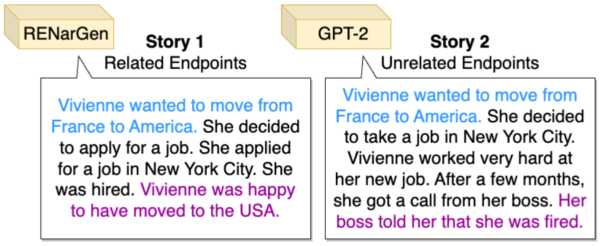

Figure 1: Stories with related start and stop sentences (Story 1, generated by RENARGEN) provide better narrative closure than stories with unrelated endpoints (Story 2, generated by GPT-2 baseline).

We refer to the first sentence as the *start*, the last sentence as the *stop*, and the start/stop sentence pair as *endpoints*. Narrative closure can be achieved via *related endpoints*, which may be operationalized with various methods, the most common of which is semantic relatedness. Endpoints are semantically related (Mohammad, 2008; Abdalla et al., 2023) if they resemble each other w.r.t. elements like theme, character, action, place. Figure 1 illustrates this idea with two stories: Story 1 has related endpoints sharing semantic commonalities that complete themes introduced in the start (e.g., protagonist → Vivienne, action → moving, and place → USA); Story 2 has unrelated endpoints with fewer semantic similarities; the stop introduces new themes without satisfactorily fulfilling the initial narrative thought. To a reader, stories like Story 1 are more "closed" than stories like Story 2.

RENARGEN (Figure 2) is a scheme that produces stories with closure using neural language models (LMs) and large language models (LLMs) by (1) generating related endpoints given the start and (2) infilling middle sentences given left and right contexts. We approach these two challenges differently for LMs versus for LLMs. For the first challenge for LMs, we use semantic relatedness

[1]Code/resources: https://github.com/adbrei/RENarGen

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers), pages 101–112

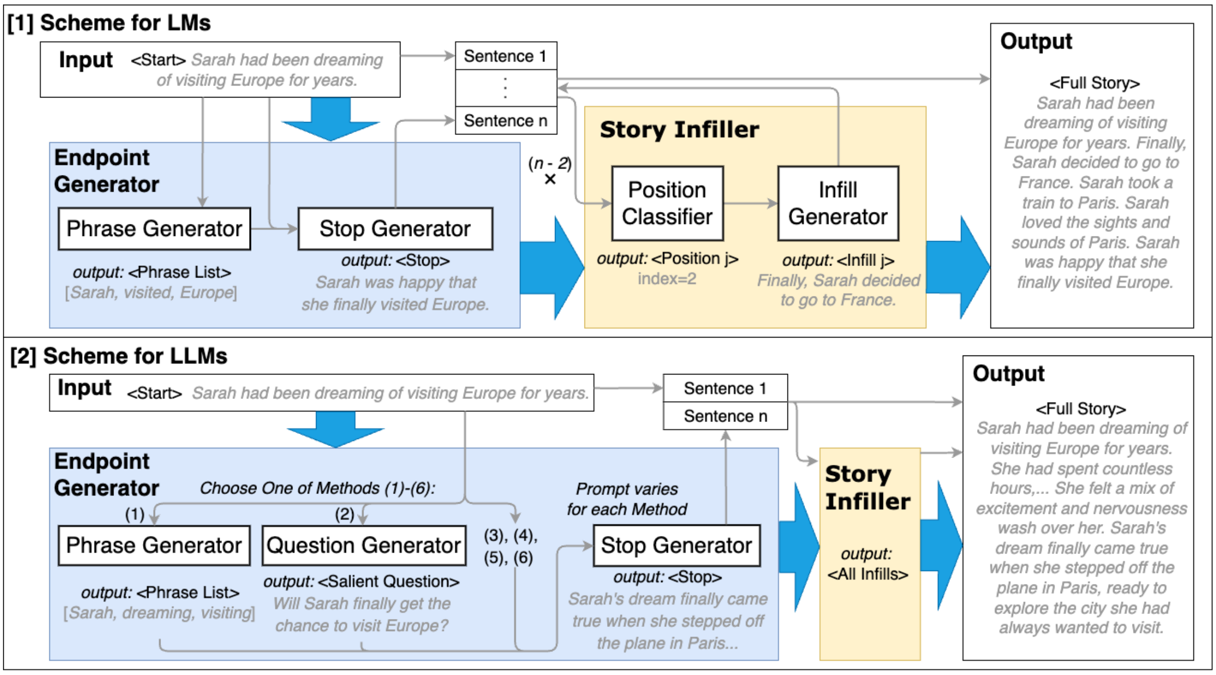

Figure 2: Proposed RENARGEN framework. **Box 1**: Scheme for LMs. Given input start, the Phrase Generator produces a phrase list of relatable words; using this list, the Stop Generator outputs the stop. The Story Infiller infills middle sentences by iteratively determining the next best location for a new sentence and generating a sentence. A sample step-by-step story generation is given in Appendix A. **Box 2**: Scheme for LLMs. Given input start, Endpoint Generater chooses one of six methods to generate the stop. The Story infiller uses the start and stop to generate all infills. After data cleaning, all components are concatenated into the final full story.

to encourage narrative closure. We generate a phrase list (salient words/phrases from the start) to emphasize narrative aspects that should be addressed in the stop. For LLMs, we experiment with different single/multi-prompting methods that address bookending with more sophisticated definitions of relatedness for narrative closure. For the second challenge for LMs, we propose an interative infilling method, inspired by story-completing techniques described in Narratology (Zemliansky, 2020), that considers both left and right contexts and generates any number of sentences in a reasonable order. While adding sentences left-to-right is a common method of expanding a story, infilling is also a bonafide method: the basic intuition is to find two consecutive sentences between which additional story material is needed. Infilling imitates human writers who add sentences to earlier locations where they determine additional information is necessary (Zemliansky, 2020; Flower and Hayes, 1981; Van Waes and Schellens, 2003; Milligan, 2017; Turner and Katic, 2009). Our method is different from previous works using an automatic bidirectional attention strategy (Devlin et al., 2018; Ippolito et al., 2019; Gu et al., 2019; Song et al., 2019; Zhu et al., 2019; Wang et al., 2020; Joshi et al., 2020; Donahue et al., 2020) that require the infill to have fixed length, require knowledge of the infill location at the beginning of inference, and/or are not easily iterable. For LLMs, the Story Infiller generates all infills at one time.

Through piece-wise narrative generation, RENARGEN offers user interactivity. For example, for LMs the user can control the generated stop sentence by editing the phrase list.

RENARGEN uses both LMs and LLMs because both have their strengths. LMs are more accessible with predictable output format but are less coherent. LLMs produce higher quality generation but are less accessible and require more computing power. See Appendix B.1 for further discussion.

Automatic and human evaluations indicate RENARGEN outperforms baselines with stories that feel more complete. Our contributions are:

- We present the first study of how related endpoints affect narrative generation with an early outlook on how the "good writing practice" of bookending impacts language modeling.
- We propose RENARGEN, a paradigm adaptable to LMs and LLMs and that produces narratives with related endpoint sentences using a novel infilling strategy.

- We conduct automatic and human evaluations to show that the stories generated by RENARGEN have related endpoints that help with narrative closure and that improve coherence.

2 RENARGEN

Given start sentence, s_1, RENARGEN generates a story $S = \{s_1, s_2, \ldots, s_n\}$ where n is the total number of sentences and the endpoints s_1 and s_n are related. RENARGEN has two main components: the Endpoint Generator and Story Infiller.[2]

2.1 Endpoint Generator for LMs

Given start, s_1, the Endpoint Generator produces a related stop, s_n. This component has a *Phrase Generator* that generates a phrase list of relatable tokens from the start and a *Stop Generator* that generates the stop for the given start incorporating the phrase list.[3] See Figure 2 for example generations.

The Phrase Generator is an LM that autoregressively outputs a phrase list, $l = [t^1, \ldots, t^r]$ where each t^i is a token with the potential to relate s_1 and a future s_n. For our experiments, we use GPT-2 (Radford et al., 2019) fine-tuned on pairs of start sentences, s_1, and their corresponding phrase lists, l, extracted from our dataset (Sec. 3.1). For extracting phrase lists, we measure similarity of each start token embedding with each stop token embedding via cosine similarity of BERT (uncased) (Devlin et al., 2018) embeddings and extract stop tokens with similarity greater than threshold, γ.[4]

The Stop Generator autoregressively accepts the concatenation of s_1 and l and outputs s_n. We use GPT-2 fine-tuned on triples of starts, extracted phrase lists, and stops of stories from our dataset.

2.2 Story Infiller for LMs

The Story Infiller generates the sentences between the endpoints, $\{s_2, s_3, \ldots, s_{n-1}\}$. It does not infill the sentences in a left-to-right manner and instead dynamically decides where to infill a sentence by determining where context is missing. The Story Infiller consists of two models: a *Position Classifier* and an *Infill Generator*.

The Position Classifier analyzes all positions between consecutive sentences in the story so far and decides the infilling position, $i \in \{2, 3, \ldots, n-1\}$. The i is the index that needs the most information for the story to sound coherent. We fine-tune BERT (uncased) to predict if the story is missing a sentence at a given position. We construct positive examples by randomly masking 1-3 sentences per story in our dataset. We construct negative samples by inserting one mask token where the story is not missing any sentences. During inference, the model considers all possible infilling positions in an incomplete story, and the position with maximum probability is selected as the next infill location.

The Infill Generator generates the missing sentence, s_i. We fine-tune GPT-2 on samples with $s_1, \ldots, s_{i-1} \langle mask \rangle s_{i+1}, \ldots, s_n \langle sep \rangle s_i$ to generate s_i. We insert the generated sentence, s_i into the story, s, and repeat the infilling process until $n-2$ sentences are infilled. We note n is a flexible threshold specified by the user.

Through this process, the Story Infiller (1) does not depend on a specific location for infill, (2) considers both left and right contexts, and (3) considers all sentences in the context, where n may be an arbitrary number of sentences set by the user. Appendix D.1 shows examples of stories of varying n-sentence lengths generated by RENARGEN.

2.3 RENARGEN for LLMs

Shown in Box 2 of Figure 2, RENARGEN for LLMs also uses an Endpoint Generator and Story Infiller. The Endpoint Generator prompts a pre-trained LLM using one of a set of methods specified by the user during inference to generate endpoints based on various definitions of bookending for narrative closure from narratology theory:

(1) Prompt for a phrase list p from the start and generate a corresponding stop (parallels the structure of RENARGEN-LM);

(2) Prompt for a "related" stop given the start and the LLM's pre-trained knowledge of sentence relatedness;

(3) Prompt for the salient narrative question introduced by the start and generate a stop that answers the question. Addresses the erotetic definition of story closure (Carroll, 2007) by concluding the salient narrative question;

(4) Generate a stop with the same character, related action, and/or location as the start. Seeks stricter control for specific narrative elements and follows the "matching ending" technique (Novakovich, 2008) for narrative closure;

[2] See Appendix C for additional implementation details.

[3] We determine that by using a phrase list of related words, the generated stop incorporates aspects from the start, thereby addressing themes and potentially questions raised by the start.

[4] For this task, we use $\gamma = 0.7$, a high threshold to ensure the phrase list contains only the most relevant related tokens.

Models	Lexical Overlap	Cosine Sim.	Syntax Sim.	Distinct n-grams	BLEU
RENARGEN-LM	**0.329**±0.136	**0.653**±0.121	**0.594**±0.110	**0.524**	**3.35**±0.15
w/out PG	0.298±0.124	0.622±0.122	0.595±0.111	–	–
w/out PC	–	–	–	0.4346	2.93±0.16
GPT-2 Baseline	0.183±0.123	0.458±0.143	0.533±0.112	0.420	3.14±0.15
RENARGEN-LLM-7b (1)	0.509±0.081	0.829±0.052	0.214±0.026	**0.773**	1.622±0.936
RENARGEN-LLM-7b (2)	0.562±0.084	0.844±0.055	0.208±0.033	0.761	**1.661±0.971**
RENARGEN-LLM-7b (3)	0.572±0.091	0.847±0.055	0.212±0.054	0.758	1.572±0.791
RENARGEN-LLM-7b (4)	**0.589**±0.103	**0.854**±0.059	**0.252**±0.093	0.748	1.579±0.897
RENARGEN-LLM-7b (5)	0.520±0.096	0.795±0.087	0.203±0.022	0.767	1.578±0.942
RENARGEN-LLM-7b (6)	0.565±0.077	0.844±0.052	0.205±0.023	0.770	1.766±1.252
w/out EG & SI	0.491±0.133	0.749±0.091	0.217±0.055	0.763	1.649±1.160
Llama-7b-chat Baseline	0.494±0.093	0.772±0.074	0.207±0.048	0.762	1.613±1.432
RENARGEN-LLM-70b (1)	0.522±0.069	0.842±0.040	0.199±0.005	**0.798**	**2.415**±1.299
RENARGEN-LLM-70b (2)	0.526±0.071	0.844±0.053	0.195±0.015	0.791	1.797±1.526
RENARGEN-LLM-70b (3)	**0.594**±0.0753	**0.870**±0.045	0.199±0.005	0.787	1.576±0.916
RENARGEN-LLM-70b (4)	0.523±0.0872	0.061±0.566	0.192±0.022	0.783	1.877±1.476
RENARGEN-LLM-70b (5)	0.512±0.0844	0.064±0.576	0.194±0.017	0.783	2.239±2.102
RENARGEN-LLM-70b (6)	0.512±0.0935	0.083±0.398	0.192±0.019	0.782	2.244±1.874
w/out EG & SI	0.526±0.085	0.805±0.082	0.192±0.020	0.795	2.351±1.385
Llama2-70b-chat Baseline	0.476±0.071	0.772±0.066	0.199±0.005	0.783	2.140±1.407

Table 1: Automatic evaluation of endpoint relatedness (first 3 cols) and overall quality (last 2 cols). Indented models are ablation studies. Bold text indicates statistical significance, $p < 0.05$ (Dror et al., 2018). Results for LLMs were conducted on a subset of the data for resource and computational cost considerations. RENARGEN generates more coherent and closed stories.

	Rel.	Clos.	Coh.	Pref.
RENARGEN-LM	**0.63**	**0.47**	**0.62**	**0.66**
GPT-2	0.20	0.18	0.21	0.21
Tie	0.17	0.35	0.17	0.13
RENARGEN-LLM-7b	**0.58**	**0.56**	**0.55**	**0.56**
Llama-7b	0.39	0.43	0.41	0.43
Tie	0.03	0.01	0.04	0.01
RENARGEN-LLM-70b	**0.80**	**0.56**	**0.56**	**0.56**
Llama-70b	0.20	0.44	0.44	0.44
Tie	0.0	0.0	0.0	0.0

Table 2: Human evaluation of RENARGEN vs baselines, showing humans prefer RENARGEN-generated stories. Bold text indicates statistical significance, $p < 0.05$.

(5) Generate a stop that entails the start. Hence, the truth of the stop logically leads to the truth of the start, resulting in semantically close endpoint sentences;

(6) Generate a stop entailed by the start. Hence, the truth of the start logically leads to the truth of the stop, resulting in semantically close endpoint sentences.

See Appendix E for additional prompting details.

The Story Infiller receives the start and generated stop and infills all sentences (an arbitrary number or a specified n) in a left-to-right manner. For our experiments, we generate 5-sentence stories with pre-trained Llama2 models (Touvron et al., 2023). Examples of longer generations are given in Appendix D.2

3 Empirical Evaluation

3.1 Dataset

We use the ROCStories corpus (Mostafazadeh et al., 2016), a collection of 5-sentence human-written stories. For RENARGEN for LMs, we combine Spring 2016 and Winter 2017 sets and obtain 98,161 stories which are split 80:20 for training and validation. For evaluation, we use the 3742 stories from Cloze Spring 2016.

3.2 Automatic Evaluation

For LMs, we compare RENARGEN with a baseline GPT-2 fine-tuned on all training samples in the ROCStories corpus; at runtime, given the start, the baseline generates a corresponding five-sentence story in a left-to-right manner. For LLMs, the baselines are Llama2-7b and Llama2-70b, where prompts do not specify endpoint relatedness. We evaluate endpoint relatedness and overall quality of generated stories. Table 1 shows the results.

Evaluating endpoint relatedness (or narrative closure) is challenging. In this work, we quantify it automatically with five metrics. We compute *Lexical overlap* via Dice Coefficient (Saad and Kamarudin, 2013), *Cosine similarity* with Sentence-BERT embeddings (Reimers and Gurevych, 2019)[5], and

[5]Endpoint relatedness is measured via cosine similiarity of start and stop sentence embeddings generated by Sentence-BERT fine-tuned on *STR-2022* (Abdalla et al., 2023), a dataset of 5,500 English sentence pairs with relatedness scores.

Syntax similarity with FastKASSAM (Chen et al., 2023; Boghrati et al., 2018) that uses a label-based tree kernel. For overall quality, we compute the average of all *distinct n-grams* ($n = \{1 \dots 5\}$) for measuring repetition and lexical creativity, and *comparison against reference stories* with BLEU score (Papineni et al., 2002; Post, 2018). For all of these measures, a higher value is better.

From the endpoint relatedness scores, we see RENARGEN is capable of generating more related endpoints than the baselines. From the overall quality scores, we see RENARGEN generates more coherent stories with more diverse content.

We conduct ablations to test the importance of various components of RENARGEN. For RENARGEN-LM, the ablation experiments remove (1) the Phrase Generator by generating the stop directly from the start and (2) the Position Classifier by adding each new sentence to a randomized position. For RENARGEN-LLM, the ablation removes the Story Infiller by simultaneously generating the entire story from left-to-right with a specified stop related to the start. Our experiments indicate the Phrase Generator and Position Classifier for LMs and the Endpoint Generator and Story Infiller for LLMs are important.

3.3 Human Evaluation of Quality

We conducted human evaluations on the Amazon Mechanical Turk (AMT) platform. We randomly sampled generated stories and performed pairwise comparisons between RENARGEN and corresponding baselines. Presentation order of the stories was randomized. Evaluators were asked to select the story with the better related endpoints, sense of closure, coherency, and overall quality. The evaluators were asked to not consider other criteria while evaluating on a specific criterion, except when judging overall quality. Sample story pairs and the instructions are shown in Appendix F and G. We limited the task to USA-based master workers with 98% approval rates and more than 5000 approved HITs. We evaluated 100 story pairs per comparison. Results (see Table 2) show evaluators preferred RENARGEN stories across all criteria. This indicates RENARGEN can produce coherent narratives that are better at providing a sense of closure to the human reader.

3.4 Human Evaluation of Interactivity

We conducted a human evaluation of interactivity with RENARGEN-LM. We asked 8 in-house testers (native English speakers with minimum higher education degree of Bachelor's) to edit phrase lists generated by the Phrase Generator on 50 unique starts. Evaluators had unlimited editing attempts per input. At the end of each interaction, they answered (1) whether or not they could generate better stories than the initial RENARGEN stories via interactivity, and (2) how useful they found the feature of editing phrase lists. For the majority of the interactions (80%), users found that the ability to control the phrase list enabled them to generate better stories than the automatically generated RENARGEN stories, and 62.5% users ranked the usefulness of interactivity as 4 on a 0-5 scale (5 being most useful). On average, users tried 3 unique phrase lists per start. These results indicate the phrase list is important for story generation and users enjoy having the ability to control this aspect.

4 Conclusion

We present RENARGEN to automatically generate stories with related endpoints via various methods of bookending. RENARGEN for LMs uses semantic relatedness and RENARGEN for LLMs uses several forms of semantic relatedness, erotetic closure, "matching ending," or entailment to guide the generation of related endpoints. We empirically demonstrate RENARGEN produces more closed, satisfying, and coherent narratives than corresponding baselines. We also show that users find RENARGEN-LM's element of controllability useful. Through our experiments and corresponding human evaluations for pair-wise preference, we further demonstrate the applicability of narratology theory for improved automatic generations and the importance of narrative closure for satisfying narratives.

5 Acknowledgement

The authors are thankful to the anonymous reviewers, and to Somnath Basu Roy Chowdhury, Haoyuan Li, and Anvesh Rao Vijjini for their constructive comments. We thank Amartya Banerjee, Nathan Brei, ML Brei, William Brei, Denali Dahl, Katharine Henry, Benjamin Linford, Vaidehi Patil, Marlus Pedrosa, Simantika Roy, Lindsey Whitlow for their feedback on evaluating interactivity and/or using RENARGEN.

6 Limitations

Since the generative model components of RENARGEN have been fine-tuned solely on datasets of stories written in English, RENARGEN can only generate text in English. For similar reasons, due to its training data, it is also limited to generating story narratives.

7 Ethics Statement

The GPT-2 components of RENARGEN are fine-tuned on the ROCStories corpus, a dataset which has been shown to have gender bias (Huang et al., 2021). As such our system might replicate or amplify this bias and other potential biases in the training dataset.

References

Mohamed Abdalla, Krishnapriya Vishnubhotla, and Saif M. Mohammad. 2023. What makes sentences semantically related: A textual relatedness dataset and empirical study. In Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, Dubrovnik, Croatia. Association for Computational Linguistics.

Giuliana Adamo. 1995. Beginnings and endings in novels. New Readings.

Amal Alabdulkarim, Siyan Li, and Xiangyu Peng. 2021. Automatic story generation: Challenges and attempts. In Proceedings of the Third Workshop on Narrative Understanding, pages 72–83, Virtual. Association for Computational Linguistics.

Reihane Boghrati, Joe Hoover, Kate M Johnson, Justin Garten, and Morteza Dehghani. 2018. Conversation level syntax similarity metric. Behavior research methods, 50(3):1055–1073.

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.

Faeze Brahman, Alexandru Petrusca, and Snigdha Chaturvedi. 2020. Cue me in: Content-inducing approaches to interactive story generation. In Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, pages 588–597.

Noël Carroll. 2007. Narrative closure. Philosophical studies, 135:1–15.

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

Snigdha Chaturvedi, Dan Goldwasser, and Hal Daume III. 2016. Ask, and shall you receive? understanding desire fulfillment in natural language text. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 30.

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, pages 1603–1614, Copenhagen, Denmark. Association for Computational Linguistics.

Maximillian Chen, Caitlyn Chen, Xiao Yu, and Zhou Yu. 2023. FastKASSIM: A fast tree kernel-based syntactic similarity metric. In Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 211–231, Dubrovnik, Croatia. Association for Computational Linguistics.

Somnath Basu Roy Chowdhury, Faeze Brahman, and Snigdha Chaturvedi. 2021. Is everything in order? a simple way to order sentences. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 10769–10779.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2492–2501, Online. Association for Computational Linguistics.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: Long papers), pages 1383–1392.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. arXiv preprint arXiv:1805.04833.

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.

Linda Flower and John R Hayes. 1981. A cognitive process theory of writing. College composition and communication, 32(4):365–387.