STORYTELLER: An Enhanced Plot-Planning Framework for Coherent and Cohesive Story Generation

Anonymous ACL submission

Abstract

Stories are central to human culture, serving to share ideas, preserve traditions, and build connections. Automatic story generation, a key advancement in AI, offers new possibilities for creating personalized content, exploring creative ideas, and enhancing interactive experiences. However, existing methods struggle to maintain narrative coherence and logical consistency. This disconnect undermines the overall storytelling experience, indicating a need for substantial improvements. Inspired by human cognitive processes, we introduce STORYTELLER, a novel approach designed to improve the coherence and consistency of automatically generated stories. STORYTELLER introduces a plot node structure based on subject-verb-object (SVO) triplets, which captures essential story events and ensures a consistent logical flow. Unlike previous methods, STORYTELLER integrates two dynamic modules—the STORYLINE and narrative entity knowledge graph (NEKG)—which interact continuously with the story generation process. This integration results in more cohesive and immersive narratives. Extensive experiments show that STORYTELLER significantly outperforms existing approaches, achieving 80.33% average win rate and leading across multiple evaluation dimensions, including creativity, coherence, engagement, and relevance. Our code and data will be made publicly available upon publication.

1 Introduction

"Stories make us more alive, more human, more courageous, more loving."
— Madeleine L'Engle

Stories have always been integral to our society, serving as a means to share knowledge, entertain, and connect on a deeper emotional level (Cupitt, 1991). From ancient oral traditions to modern digital media, storytelling shapes our understanding of the world and our connections with each other. With the rise of artificial intelligence, the automatic generation of stories (Alhussain and Azmi, 2021; Alabdulkarim et al., 2021) emerges as a promising field in natural language processing (NLP), offer personalized educational content, and enhance user experiences in interactive systems. While large language models (LLMs) (Achiam et al., 2023; Touvron et al., 2023; Zhang et al., 2022) excel at generating stylistically fluent text, they often struggle to maintain coherence and logical consistency, especially when it comes to maintaining plot structure and thematic continuity throughout the story.

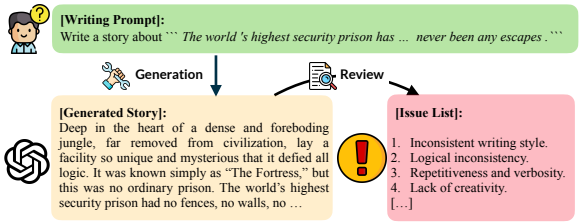

Figure 1: Story generation still faces challenges like inconsistent style, logic issues, lack of creativity and so on.

Existing hierarchical and event-based generation methods provide high-level outlines to guide the story development (Fan et al., 2018; Yao et al., 2019; Rashkin et al., 2020a). However, these methods often generate chapters or events independently, leading to plot inconsistencies and a lack of narrative cohesion. For example, independently generated story segments may cause sudden shifts in character motivations or abrupt plot changes that disrupt the overall reading experience (Yao et al., 2019). Such limitations hinder the ability of these methods to create engaging and immersive stories.

To address these challenges, we draw inspiration from the cognitive process of human writers (Lamott, 1995; Cook, 2011). Consider a writer carefully crafting a complex story; they contin-

uously recall past events, evaluate how new developments align with previous ones, and create new content that seamlessly fits into the existing narrative (Clark, 2008; Brande, 2013). This cognitive loop—Recalling, Evaluating and Creating—ensures that the story evolves naturally, with new events logically building on past developments.

Building on this insight, our proposed framework, STORYTELLER, introduces an innovative approach to story generation by enhancing plot-planning mechanisms. Unlike previous methods, STORYTELLER facilitates continuous interaction between plot points during the generation process, ensuring a more cohesive narrative structure. A central feature of our approach is the refined plot node mechanism, which tracks key events and character interactions using subject-verb-object (SVO) triplets with storyline and narrative entity knowledge graph. This design allows STORYTELLER to maintain logical flow and thematic consistency throughout the story. By leveraging a refined plot node mechanism that tracks key events and character interactions over time, STORYTELLER provides greater control over plot progression, enhancing both consistency and thematic richness.

Highlights of our contributions are as follows:

- We propose an innovative plot-planning framework through a novel plot node structure using subject-verb-object (SVO) triplets. This design captures critical story events, ensuring logical flow and consistency throughout the narrative, which is crucial for maintaining a coherent storyline.
- We design the modules: STORYLINE and narrative entity knowledge graph (NEKG) interact dynamically with the story generation process, ensuring continuous refinement of plot structure and event coherence across different stages.
- Our experiments demonstrate that our approach significantly outperforms existing models in generating stories that are not only coherent but also more engaging and creative with better quality.

2 Related Work

Large Language Model. The emergence of large language models (LLMs) has fundamentally transformed the field of natural language processing (NLP), marking a significant milestone in its development (Vaswani et al., 2017; Brown, 2020; Zhang et al., 2023). Early models, like BERT (Devlin et al., 2019), Transformer (Vaswani et al., 2017) lay the groundwork for more advanced architectures. Models such as GPT-4 (Achiam et al., 2023), Llama-3 (AI@Meta, 2024), and Qwen (Bai et al., 2023) are renowned for their advanced capabilities and are increasingly serving as foundational platforms for a wide range of applications. Their ability to generate coherent, and contextually relevant text has led to significant advancements in tasks such as machine translation, text summarization, and conversational agents, impacting numerous fields including healthcare, education, and finance.

Automatic Story Generation. In previous studies, many methods have employed hierarchical generation strategies for story creation (Fan et al., 2018; Fang et al., 2021), where a high-level outline is first produced, followed by the generation of individual chapters or paragraphs based on this outline. This framework is designed to guide storytelling in a structured way, helping models generate narrative content more systematically (Yu et al., 2020; Yao et al., 2019).

However, these methods often struggle to control the storyline's progression because each chapter is typically generated independently, lacking visibility into and relevance with other parts. This lack of interaction can lead to plot inconsistencies, incoherent character development, or missing thematic elements, ultimately diminishing overall narrative coherence. To address these issues, recent research explores enhancing interaction among different parts of the story during generation (Chen et al., 2023; Zhou et al., 2023; Wang et al., 2023). Some approaches introduce global context representations (You et al., 2023; Castricato et al., 2021), utilize memory networks (Park et al., 2024; Han et al., 2022), or employ cross-attention mechanisms to ensure each section can reference and incorporate developments from other parts (Guan et al., 2022; Tang et al., 2022; Rashkin et al., 2020b). These advancements aim to enhance control over the storyline and promote continuity, resulting in more cohesive and compelling narratives.

Prompting. Prompting involves a user providing a text input to the model, which generates a response from the language model based on the given input(Liu et al., 2023). A key feature contributing to the versatility of LLMs is in-context learn-

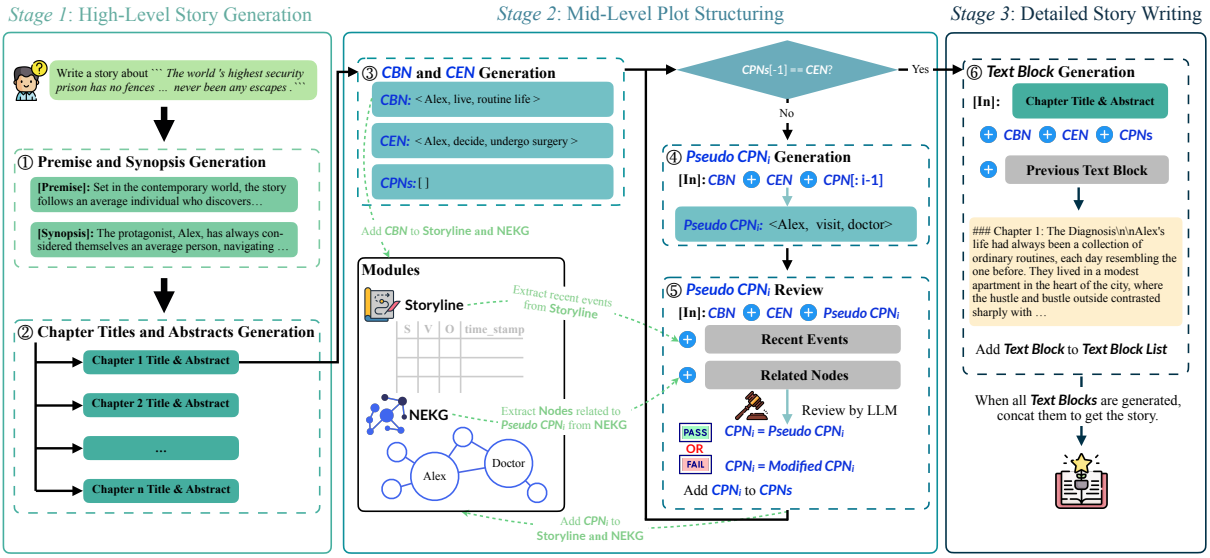

Figure 2: Overview of STORYTELLER. The method is divided into three stages. This figure illustrates the complete process by STORYTELLER transforms user input into the final story. Additionally, it highlights the information interaction between various stages and both the STORYLINE and NEKG.

ing, which enables these models to perform tasks based solely on the information provided within a prompt, without requiring explicit parameter updates (Dong et al., 2022; Wei et al., 2022). This capability grants LLMs exceptional flexibility and adaptability, allowing them to generalize across a wide range of tasks with minimal additional training (Levine et al., 2022; Zhao et al., 2021).

3 Methodology

This section presents the framework of STORYTELLER and the relationships among its various processes. It begins with an introduction to three key modules (§3.1), followed by a top-down workflow detailing *Stage 1* (§3.2), *Stage 2* (§3.3), and *Stage 3* (§3.4).

3.1 Modules

NODES. In STORYTELLER, the most critical element is the NODE. Each NODE represents either a subject-verb-object (SVO) or a subject-verb (SV). To unify into triplets, SV are converted into subject-verb-subject (SVS) (Equation 2).

$$\text{NODE(SVO)} =< \text{Subject}, \text{Verb}, \text{Object} > \quad (1)$$

$$\text{NODE(SV)} =< \text{Subject}, \text{Verb}, \text{Subject} > \quad (2)$$

NODES link STORYLINE and NEKG with the story generation process, facilitating high-level information. Furthermore, we classify NODES based on their position in the chapter into *CBN*, *CPN* and *CEN* as shown in Table 1.

NODE	Description
CBN	The start node of a chapter
CPN	Plot node within the chapter, driving the story forward
CEN	The end node of a chapter

Table 1: Three NODE types: *CBN*(Chapter Begin Node), *CPN*(Chapter Plot Node) and *CEN*(Chapter End Node). These NODES represent stages in the narrative structure of a chapter.

STORYLINE. STORYLINE is used to store the NODES that have been generated so far, tracking of the development of the story's plot.

Each NODE is associated with a `time_stamp` that indicates when the NODE occurs. This allows us to establish the temporal sequence between different NODES, ensuring to accurately determine the order in which events unfold. By capturing these chronological relationship, the Storyline effectively tracks the progression of the narrative, providing a clear and structured record of the evolving plot. This feature is crucial for maintaining story coherence and understanding how different events and actions are interconnected over time.

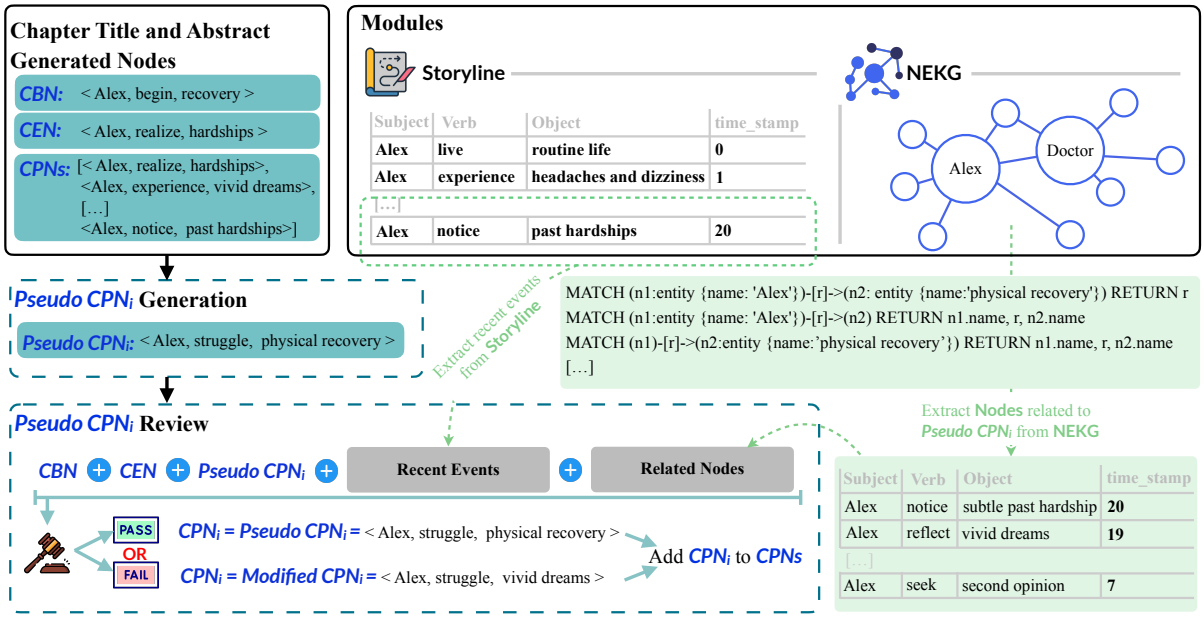

Figure 3: *Pseudo CPN* Generation and Review. Specific example is used to provide a detailed description of the process. Additionally, it is clearly explained how the STORYLINE and NEKG provide relevant information to aid in the generation of *CPN*.

NEKG. Narrative entity knowledge graph (NEKG) represents entities and their interrelationships in a structured, graph-like format. It consists of nodes and edges, where nodes represent entities or concepts (such as people, places, objects), and edges denote the relationships between these entities. NEKG effectively illustrates interactions between characters, the circulation of objects, and the logical sequence of events within the story.

NEKG aids both systems and readers in intuitively understanding the key elements driving the story's progression and uncovering hidden relational chains. We implement NEKG using Neo4j[1] and each generated story has a corresponding NEKG.

In addition, NEKG offers significant extensibility. It can not only describe the current storyline but also support future plot generation and reasoning. By leveraging rich associative data and logical relationships, the graph enables the expansion of narratives based on existing information, facilitating the creation of new plotlines or character development paths. This capability can aid not only in story analysis but also in inspiring and guiding future creative endeavors.

[1] https://neo4j.com/

3.2 *Stage 1*: High-Level Story Generation

In this stage, STORYTELLER begins by generating the story at a high-level. First, it consolidates the user-provided information to generate the premise and synopsis. Next, to ensure the coherence and smooth flow, it generates the titles and abstracts for all chapters. The prompt template is provided in Appendix B.1.

Premise and Synopsis Generation. This phase aims to transform the user's input into two key components: a concise premise and a detailed synopsis. The input is analyzed to extract its core elements, which are used to generate the corresponding components. The premise focuses on describing the story's era, setting, and societal context, while the synopsis encompasses the main plot, character dynamics, and key narrative twists.

Chapter Titles and Abstracts Generation. The objective of this phase is to structure a long-form story into chapters based on its premise and synopsis, while generating appropriate titles and detailed content abstracts for each chapter. This process enables the model to establish a comprehensive framework of the story, ensuring that the subsequent division of chapters and content creation remain consistent with the story's background and core narrative. Each chapter is assigned a relevant

title and a detailed summary, describing key events and elaborating on character interactions and plot developments to advance the storyline.

3.3 *Stage 2*: Mid-Level Plot Structuring

In this stage, STORYTELLER generates NODES for each chapter. The Chapter Begin Node (*CBN*) and Chapter End Node (*CEN*) are created first. After that, the Chapter Plot Nodes (*CPN*) are generated sequentially in two steps. The prompt template is provided in Appendix B.2.

***CBN* and *CEN* Generation.** The task here is to generate *CBN* and *CEN* for each chapter. The steps involve reviewing the title and abstract of the chapter. The generated *CBN* and *CEN* must ensure logical coherence and smooth transition into other parts of the story, as these two NODES represent the begginning and conclusion of the chapter. Therefore, we incorporate information not only from the current chapter but also from the preceding and following chapters.

***Pseudo CPN* Generation and Review.** Thanks to the previous module design, the information of generated story can be structurally stored, providing a foundation for subsequent *CPN* generation to ensure plot quality. Although STORYLINE and NEKG contain vast amounts of data, limitations imposed by the context length and cost of LLMs require selecting only a portion of data for generating. This rasises a new challenge: "How to extract relevant information from extensive data?"

To address the issue, *Pseudo CPN* is generated based on existing *CBN*, *CEN* and *CPN*. Using the *Pseudo CPN* ($S'V'O'$), relevant nodes matching the S' and O' are retrieved from NEKG, representing the most recent events between S' and O'. The nodes where the subject or object is S' or O' are also retrieved, representing recent events related to S' and O' as shown in Figure 3.

The nodes are then reviewed to assess whether the *Pseudo CPN* maintains coherence and fluency. If it passes, the *Pseudo CPN* is accepted as the *CPN*. If it fails, LLM generates a *modified CPN* based on the contextual information and *Pseudo CPN*, along with an explanation for the modification. *CPN* is then added to *CPNs*, and this process continues until the generated *CPN* aligns with *CEN*, signaling the completion of the chapter's plot node generation.

GENERATED STORY:
C1: Alex's life had always been a collection of ordinary routines, each day resembling the one ...[379 words]...
C2: Post-Surgery ReflectionsThe sterile smell of antiseptic ...[1030 words]...
...[5 Chapters]...
C7: The atmosphere in the office was charged with tension ...[1166 words]...
C8: The quiet hum of the office had returned, a stark contrast to the chaos that had ...[1060 words]...

Table 2: A heavily abridged STORYTELLER story generation from the same writing prompts in Figure 2

3.4 *Stage 3*: Detailed Story Writing

***Text Block* Generation.** In this stage, the model is required to strictly adhere to the established plot nodes (*CBN*, *CPNs*, *CEN*) to ensure coherence and consistency throughout the story. The precise plot structure serves to maintain the logical flow of events. Additionally, to ensure a consistent linguistic style across the entire narrative, we also input the previously generated text block from the preceding chapter. This approach ensures that the generated text aligns with the overall narrative style and structural requirements of the story. The prompt template is provided in Appendix B.3.

4 Experiments

4.1 Experiments Setup

Dataset. To evaluate the performance of STORYTELLER, we conduct experiments using the WRITINGPROMPTS dataset (Fan et al., 2018). WRITINGPROMPTS consists of 300K human-written stories and their corresponding prompts, sourced from Reddit[2]. The prompts cover a wide range of story themes. To balance evaluation cost and ensure comprehensive measurement, we randomly select writing prompts tagged with `[ WP ]` from the source file. A more detailed description of the WRITINGPROMPTS can be found in Appendix A.

Baselines. Prior to this, most models or methods focuse on generating shorter stories, compared to STORYTELLER. Therefore, we select for baselines as follows. The prompt template is provided in Appendix B.4.

- GPT-4o (OpenAI), the most advanced GPT

[2] www.reddit.com/r/WritingPrompts/

Model / Method	Avg Word Count	Perspectives				Overall
		Creativity	Coherence	Engagement	Relevance	
Model						
GPT-4o	1,076	50.0	50.0	50.0	50.0	50.0
Qwen2-72B-Instruct	5,173	25.0	25.0	25.5	31.3	18.3
Llama3.1-70B-Instruct	1,578	46.2	50.8	42.4	49.2	31.4
LongWriter-glm4-9b	5,191	60.2	68.3	64.0	67.3	62.7
LongWriter-llama3.1-8b	5,659	47.8	53.5	51.8	54.7	43.1
Method						
DOC v2	7,323	60.4	49.7	52.2	51.6	50.9
STORYTELLER (ours)	**7,977**	**68.8**	**68.9**	**70.1**	**74.3**	**79.1**

Table 3: Overall results of different models or method. STORYTELLER excels across all metrics, with a high average word count (7,977). Also it achieves the highest Overall score of 79.1, significantly outperforming other models. It also leads in five different perspectives. The best-performing in each metric is **in-bold**.

model of OpenAI, is a high-intelligence flagship model for complex, multi-step tasks.

- The two open-source instruct models, Qwen2-72B-Instruct (Bai et al., 2023) and Meta-Llama-3.1-70B-Instruct (AI@Meta, 2024), are also used as baselines.
- LongWriter Models (Bai et al., 2024), based on Meta-Llama-3.1-8B (Dubey et al., 2024) and GLM4-9B (GLM et al., 2024), are capable of generating over 10,000 words in the output.
- DOC v2[3], an improved version of DOC (Yang et al., 2022), consists of a detailed outliner and a detailed controller. The outliner generates a structured, multi-level outline, while the controller ensures that the generated content adheres to this outline.

Configuration. All open-source models are inferred on 8 * NVIDIA A100 GPUs using the `vllm` (Kwon et al., 2023), with `temperature` set to 0 and `max_tokens` set to 16,384 in the `SamplingParams`, thereby employing greedy decoding for inference.

For DOC v2 and STORYTELLER, GPT-4o is utilized as the engine. Additionally, in all model evaluations, the judge model employed is also GPT-4o.

Metrics. To accurately assess the performance of stories generated by STORYTELLER, evaluations are conducted from multiple aspects. First, attention is given to the length of the generated texts, measured by the **Average Word Count** metric. Next, the overall quality of the stories was assessed through overall evaluation. Additionally, more detailed assessments are made by breaking down the evaluation into five specific perspectives.

1. **Creativity.** Originality of the plot and characters.
2. **Coherence.** Clarity of the structure and flow.
3. **Engagement.** The extent to which the story captivates the reader and sustains their emotional involvement and curiosity.
4. **Relevance.** Whether the text stays aligned with the theme, prompt, or background.
5. **Overall.** Alignment with the prompt, character and plot development, reader engagement, originality, and areas for improvement.

Following the arena-hard-auto (Li et al., 2024) evaluation method, stories generated by GPT-4o are used as a baseline, and comparisons are made with other generated stories to obtain quantified scores.

We also conduct a preference evaluation using GPT-4o as the annotator to compare STORYTELLER with storylines.

At the same time, attention is given to the diversity of the generated stories, as it contributes to enhancing the depth and breadth of the narrative, making the plot, characters, and setting more vibrant, while avoiding monotony. The diversity evaluation is conducted using the following metrics. **Diverse Verbs** (Fan et al., 2019), obtained by analyzing verbs within the text, is employed to

[3] https://github.com/facebookresearch/doc-storygen-v2

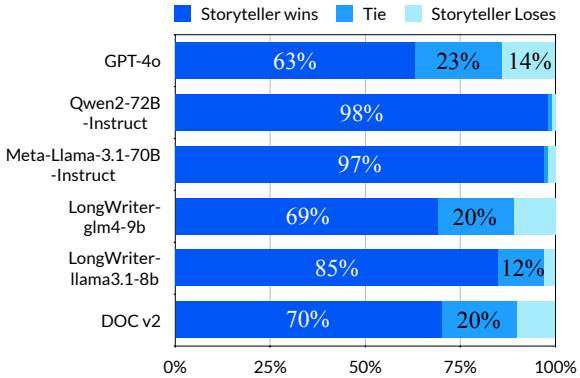

Figure 4: Preference evaluation using GPT-4o as the annotator.

reflect the diversity of actions. The more varied the verbs, the richer the generated content is in terms of action diversity. Distinct-n (Li et al., 2016) is often used to evaluate the text diversity. However, it does not account for the impact of text length. As text length increases, the occurrence of repeated n-grams may also increase, leading to a decline in the Distinct score. To provide a more equitable assessment of diversity, we introduce the factor of text length and propose **DistinctL-n**, as defined in Equation 3. This adjustment allows DistinctL to provide a more equitable assessment of diversity in longer texts.

$$\text{DistinctL} - \text{n} = \frac{\text{unique n-grams}}{\text{total n-grams}} \times (1 + \log(\text{word_count})) \quad (3)$$

4.2 Comprehensive Evaluation

Directly asking a LLM to rate generated stories from different aspects can lead to unstable results. This is due to the high variability of the stories, coupled with the fact that the generated stories are often lengthy, making it difficult to provide effective assistance to the model through few-shot in the prompt. Drawing inspiration from Li et al. (2024), we select GPT-4o as the reference model and compared stories generated by other models against it across multiple aspects. Score is defined as $\frac{1}{N}\sum_{\{i,j\}}(f(\pi_j < \pi_i) - O(\pi_j < \pi_i))^2$. To ensure that the model accurately understands the comparison criteria for each aspect, we develop a detailed checklist for each evaluation dimension, enabling the LLM to conduct comparative assessments effectively.

As shown in Table 3, STORYTELLER performs exceptionally well across various aspects. Its average word count is 7,977. Besides STORYTELLER demonstrates a significant advantage across all five evaluated dimensions—Creativity, Coherence, Engagement, Relevance, and Overall Performance. It achieves the highest Overall score of 79.1, outperforming all other models, none of which exceed 65 points. This clearly indicates STORYTELLER's superior capacity to generate high-quality, well-rounded narratives.

For Creativity, STORYTELLER scores 68.8, well above the second-best model LongWriter-glm4-9b (60.2). This reflects Storyteller's ability to produce imaginative and innovative narratives. In terms of Coherence, both Storyteller (68.9) and LongWriter-glm4-9b (68.3) perform similarly, indicating strong abilities to maintain logical and well-structured storytelling. These models generate content that flows smoothly and remains consistent, while other models lag behind in this area. STORYTELLER also excels in Engagement, scoring 70.1 compared to LongWriter-glm4-9b's 64, showing its ability to create stories that captivate the audience and sustain their interest. In the Relevance category, STORYTELLER leads again with 74.3, highlighting its capability to generate content that closely aligns with the prompt and remains focused on the topic. STORYTELLER consistently outperforms all competitors, demonstrating a clear ability to deliver well-structured, engaging, and creative narratives.

It can also be observed that the average word count of Qwen2-72B-Instruct is significantly higher than that of Llama3.1-70B-Instruct; however, its subsequent metric scores are relatively lower. This is attributed to the presence of numerous repeated paragraphs in the content generated by Qwen2-72B-Instruct, which results in a lower quality of the generated stories.

4.3 Preference Evaluation

Following Zhou et al. (2024), to compare STORYTELLER with other baselines, we generate stories based on the same set of writing prompts for all systems. We then utilize GPT-4o to compare the outputs of STORYTELLER with each baseline and identify the preferred story for each comparison.

To eliminate the positional bias of the stories, after the first comparison, the stories generated are swapped in their respective positions for a second comparison. The final result is derived by aggregating the outcomes from both rounds of comparison. The prompt template is provided in Appendix B.5.

As shown in Figure 4, STORYTELLER demon-

Model / Method	DistinctL-n				Diverse
	DistinctL-2	DistinctL-3	DistinctL-4	DistinctL-5	**Verbs**
Model					
GPT-4o	2.640	2.847	2.870	2.874	0.96
Qwen2-72B-Instruct	1.572	1.888	2.004	2.064	0.83
Meta-Llama3-3.1-70B-Instruct	2.297	2.692	2.820	2.879	0.91
LongWriter-glm4-9b	2.302	2.963	3.196	3.303	0.92
LongWriter-llama3.1-8b	1.905	2.417	2.596	2.679	0.92
Method					
DOC v2	**3.469**	**3.669**	3.696	3.703	**0.97**
STORYTELLER (ours)	2.775	3.578	**3.791**	**3.855**	0.94

Table 4: Diversity results of different models or method. STORYTELLER excels across all metrics, with top rankings in diversity, particularly achieving the highest DistinctL-4 and DistinctL-5 scores. The best-performing in each metric is **in-bold**, and the second best is underlined.

strates superior performance in story generation, generation, consistently outperforming all four baselines. When compared to GPT-4o, which also serves as the annotator, STORYTELLER wins in 63% of cases, tying in 23%, and losing in 14%. For both Instruct models, the STORYTELLER demonstrates an exceptionally high win rate (>95%), indicating that the stories it generates are significantly superior to those produced by the models. Similarly, STORYTELLER outperforms LongWriter-glm4-9b in 69% of comparisons, with a 20% tie rate, indicating that while LongWriter-glm4-9b can sometimes match STORYTELLER, it still lags in overall story quality. It achieves an impressive win rate of 85% against LongWriter-llama3.1-8b, with only a 12% tie and a 3% loss rate.

Against DOC v2, STORYTELLER wins 70% of the time, with a 20% tie rate, reaffirming its robust performance across all evaluations. These results collectively highlight STORYTELLER's ability to generate more coherent, creative, and engaging stories compared to the baselines.

4.4 Diversity Evaluation in Story Generation

Table 4 presents the diversity metrics for all models and methods. Overall, STORYTELLER ranks within top two across all DistinctL-n metrcs, showcasing strong diversity in both short and long sequences. Notably, it achieves the highest scores in DistinctL-4 (3.791) and DistinctL-5 (3.855), which indicates a high degree in creativity and richness in story generation. Additionally, STORYTELLER performs comparably well to DOC v2 in the Diverse Verbs metric, indicating that it generates dynamic and varied actions in its narratives.

It can also be observed that the diversity metric scores of Qwen2-72B-Instruct are relatively low, which aligns with the previously mentioned issue of generating a large number of repeated paragraphs. In contrast, other models demonstrate varying levels of diversity. For example, GPT-4o performs reasonably well in DistinctL-2 and DistinctL-5 but does not lead in any of the categories. The LongWriter models, particularly LongWriter-llama3.1-8b, show consistently lower, suggesting limitations in maintaining diversity throughout.

5 Conclusion

In this paper, we introduce STORYTELLER, a novel approach to improving the coherence and engagement of automatically generated stories through enhanced plot-planning mechanisms. Our method addresses the challenges existing systems face in maintaining structural consistency and narrative fluidity in long-form content. By focusing on plot node management and real-time interaction between different parts of the story, STORYTELLER ensures that key events and character arcs evolve in a more controlled and logical manner.

The core motivation of our approach lies in improving the weaknesses of prior methods, which often generated disjointed or inconsistent narratives due to isolated event generation. By implementing a flexible, dynamic interaction between the STORYLINE and the narrative entity knowledge graph (NEKG), STORYTELLER achieves a more cohesive narrative flow. Our results demonstrate that this plot-planning framework leads to more engaging, creative, and coherent stories.

Limitations

Despite generating relatively high-quality stories, a substantial gap remains between these outputs and the ultimate goal of producing a complete novel. The aspiration is to create works that fully embodies the depth, complexity, and narrative richness inherent in the term "novel."

Currently, the field of story generation lacks specialized evaluation metrics capable of providing a comprehensive assessment of generated narratives. This absence represents a significant challenge, hindering the accurate evaluation and comparison of different story generation models.

Ethical Statements

This study employs large language models (LLMs) to generate and evaluate story content based on prompts. Although no personal data or animal subjects are involved, the potential ethical risks associated with content generation must still be considered. LLMs can ocasionally produce "hallucinations," generating information that is factually incorrect or irrelevant. While the focus is on story creation where non-factual narratives may not be an issue, we remain attentive to the quality and coherence of the generated text, avoiding major logical errors or inappropriate content. The evaluation process is based on automated assessments using LLMs, aimed at reducing subjective bias. However, we acknowledge the limitations of current automated evaluation tools, and therefore incorporate multi-dimensional metrics and careful judgment when analyzing the results.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

AI@Meta. 2024. Llama 3 model card.

Amal Alabdulkarim, Siyan Li, and Xiangyu Peng. 2021. Automatic story generation: Challenges and attempts. *arXiv preprint arXiv:2102.12634*.

Arwa I Alhussain and Aqil M Azmi. 2021. Automatic story generation: A survey of approaches. *ACM Computing Surveys (CSUR)*, 54(5):1–38.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Yushi Bai, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. Longwriter: Unleashing 10,000+ word generation from long context llms. *arXiv preprint arXiv:2408.07055*.

Dorothea Brande. 2013. from becoming a writer. In *Creative Writing*, pages 424–427. Routledge.

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Louis Castricato, Spencer Frazier, Jonathan Balloch, Nitya Tarakad, and Mark Riedl. 2021. Automated story generation as question-answering. *arXiv preprint arXiv:2112.03808*.

Guangyao Chen, Siwei Dong, Yu Shu, Ge Zhang, Jaward Sesay, Börje F Karlsson, Jie Fu, and Yemin Shi. 2023. Autoagents: A framework for automatic agent generation. *arXiv preprint arXiv:2309.17288*.

Roy Peter Clark. 2008. *Writing tools: 55 essential strategies for every writer*. Little, Brown Spark.

William Cook. 2011. *PLOTTO: the master book of all plots*. Tin House Books.

Don Cupitt. 1991. *What is a Story?* Trinity PressIntl.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Conference of the Association for Computational Linguistics (ACL)*.

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.

Le Fang, Tao Zeng, Chaochun Liu, Liefeng Bo, Wen Dong, and Changyou Chen. 2021. Outline to story: Fine-grained controllable story generation from cascaded events. *arXiv preprint arXiv:2101.00822*.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Mingdao Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiao Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yifan An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhen Yang, Zhengxiao Du, Zhenyu Hou, and Zihan Wang. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *Preprint*, arXiv:2406.12793.

J Guan, Z Yang, R Zhang, Z Hu, and M Huang. 2022. Generating coherent narratives by learning dynamic and discrete entity states with a contrastive framework. arxiv. *arXiv preprint arXiv:2208.03985*.

Rujun Han, Hong Chen, Yufei Tian, and Nanyun Peng. 2022. Go back in time: Generating flashbacks in stories with event temporal prompts. *arXiv preprint arXiv:2205.01898*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Anne Lamott. 1995. *Bird by bird: Some instructions on writing and life*. Vintage.

Yoav Levine, Noam Wies, Daniel Jannai, Dan Navon, Yedid Hoshen, and Amnon Shashua. 2022. The inductive bias of in-context learning: Rethinking pretraining example design. In *International Conference on Learning Representations*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

OpenAI. Hello gpt-4o | openai. `https://openai.com/index/hello-gpt-4o/`.

Kyeongman Park, Nakyeong Yang, and Kyomin Jung. 2024. Longstory: Coherent, complete and length controlled long story generation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 184–196. Springer.

Chau Minh Pham, Simeng Sun, and Mohit Iyyer. 2024. Suri: Multi-constraint instruction following for long-form text generation. *Preprint*, arXiv:2406.19371.

Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. 2020a. Plotmachines: Outline-conditioned generation with dynamic plot state tracking. *arXiv preprint arXiv:2004.14967*.

Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. 2020b. 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.

Chen Tang, Chenghua Lin, Henglin Huang, Frank Guerin, and Zhihao Zhang. 2022. Etrica: Event-triggered context-aware story generation augmented by cross attention. *arXiv preprint arXiv:2210.12463*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Yichen Wang, Kevin Yang, Xiaoming Liu, and Dan Klein. 2023. Improving pacing in long-form story planning. *arXiv preprint arXiv:2311.04459*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Transactions*

on Machine Learning Research. Survey Certification.

Kevin Yang, Dan Klein, Nanyun Peng, and Yuandong Tian. 2022. Doc: Improving long story coherence with detailed outline control. *arXiv preprint arXiv:2212.10077*.

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.

Wang You, Wenshan Wu, Yaobo Liang, Shaoguang Mao, Chenfei Wu, Maosong Cao, Yuzhe Cai, Yiduo Guo, Yan Xia, Furu Wei, et al. 2023. Eipe-text: Evaluation-guided iterative plan extraction for long-form narrative text generation. *arXiv preprint arXiv:2310.08185*.

Meng-Hsuan Yu, Juntao Li, Danyang Liu, Dongyan Zhao, Rui Yan, Bo Tang, and Haisong Zhang. 2020. Draft and edit: Automatic storytelling through multi-pass hierarchical conditional variational autoencoder. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1741–1748.

Lei Zhang, Yunshui Li, Ziqiang Liu, Jiaxi yang, Junhao Liu, and Min Yang. 2023. Marathon: A race through the realm of long context with large language models. *Preprint*, arXiv:2312.09542.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International conference on machine learning*, pages 12697–12706. PMLR.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2024. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36.

Wangchunshu Zhou, Yuchen Eleanor Jiang, Peng Cui, Tiannan Wang, Zhenxin Xiao, Yifan Hou, Ryan Cotterell, and Mrinmaya Sachan. 2023. Recurrentgpt: Interactive generation of (arbitrarily) long text. *arXiv preprint arXiv:2305.13304*.

A WRITINGPROMPTS Dataset

In this section, we provide a more detailed introduction to the WRITINGPROMPTS dataset (Fan et al., 2018), including its tags, composition, and other relevant aspects.

Prompt tags are enclosed in square brackets and serve to help users organize or filter prompts of a specific type. Reddit provides a total of 11 tags, and descriptions of each tag are shown in Table 5. More detailed descriptions of the tags can be found on the corresponding website[4].

Tag	Full Name	Description
[WP]	Writing Prompt	Basic prompts with no restrictions, providing simple ideas to inspire narrative fiction, poetry, or other forms of writing.
[SP]	Simple Prompt	A basic prompt with a title containing no more than 100 characters.
[EU]	Established Universe	For prompts set in pre-established fictional worlds, encouraging fan fiction that adds new scenarios to well-known series and characters.
[CW]	Constrained Writing	Includes constraints or limitations, allowing restrictions on words, word limits, or specific writing styles.
[TT]	Theme Thursday	Focus on weekly themed writing styles, requiring conformity to the specific theme of the week.
[PM]	Prompt Me	Encourage responding to comments as prompts, expanding creative output and writing skills by treating responses as prompts.
[MP]	Media Prompt	Use audio or visual media to inspire writing, offering an alternative to traditional text-based prompts.
[IP]	Image Prompt	Linked images to inspire stories or poems, similar to Media Prompts, but AI-generated art is not allowed.
[PI]	Prompt Inspired	Consists of standalone responses to prompts at least three days old, with the story posted in one post or in comments if too long, and must include a link to the original prompt.
[OT]	Off Topic	Off topic and not prompts, but are writing related. Not for stories.
[RF]	Reality Fiction	Focus on events that have occurred or could happen in the real world to unknown people, excluding future possibilities.

Table 5: Prompt tags in WRITINGPROMPTS and their descriptions.

Fan et al. (2018) analyze the statistics of the WRITINGPROMPTS dataset, and here we directly reference their work as shown in Table 6.

[4] https://www.reddit.com/r/WritingPrompts/wiki/how_to_tag_prompts

# Train Stories	272,600
# Test Stories	15,138
# Validation Stories	15,620
# Prompt Words	7.7M
# Story Words	200M
Average Length of Prompts	28.4
Average Length of Stories	734.5

Table 6: Statiscs of WRITINGPROMPTS dataset.

B Prompt Templates

In this section, we present a comprehensive list of all the prompt templates utilized throughout the article.

B.1 High-Level Story Generation Template

Premise and Synopsis Generation prompt:

Prompt Template

You need assistance in modifying a user input for generating long stories. The goal is to ensure the premise is concise while the synopsis is rich and detailed. The process involves the following steps:
[Requirements]:
1. Input Interpretation: The model initially interprets the user's input, extracting essential elements like background context and plot hints. This step forms the basis for crafting an engaging narrative.
2. Premise Formulation: Utilizing the input, the model constructs a succinct premise, detailing the story's era, setting, and societal backdrop. The premise will strictly outline the story's foundation and context, deliberately omitting any plot-related details.
3. Detailed Synopsis Composition: Subsequently, the model develops an extensive synopsis, targeting a minimum length of 1000 words. This synopsis will weave through the narrative's core plot, character arcs, conflicts, climax, and resolution, while emphasizing pivotal twists and character dynamics to captivate the audience.
4. Distinction Between Premise and Synopsis: The model ensures a clear demarcation between the premise and synopsis. The premise will succinctly set the stage, focusing on the world's backdrop. In contrast, the synopsis will explore the narrative's complexities and plot intricacies.
5. JSON Output Structuring: The output will be formatted in JSON, structured as follows:
“‘
{{
"premise": "<Insert story background here>",
"synopsis": "<Insert detailed story plot here>"
}}
“‘
Only output the JSON result without any other information.
Input: {user_input}

Chapter Titles and Abstracts Generation prompt:

Prompt Template

Given a story premise and synopsis, your task is to plan the chapters of a long story. The premise will provide background information about the story, including details about the era, location, environment, and social context. The synopsis will describe the main plot, character development, conflicts and their resolutions, climax, and ending of the story.
Follow these steps to plan the chapters:
1. Begin by thoroughly reading and understanding the premise and synopsis. Identify key elements of the story, such as important characters, settings, and conflicts.
2. Use your imagination to conceive a logical sequence of story development based on the identified story elements. Think about how the plot will unfold and the order in which events will occur.
3. Break down the story sequence into several chapters, ensuring that each chapter contains a key part of the story. Consider the pacing and flow of the narrative as you divide it into chapters.
4. For each chapter, create a title and write a detailed description of the chapter's content. The description should include major events, character interactions, and plot developments within the chapter.
5. After writing the initial chapter summaries, expand each summary to include specific plot details. Enrich and refine the content while ensuring coherence in the story's development. Add more depth to the events, interactions, and developments within each chapter. The content of each chapter should be at least {chapter_abstract_num} words long.
Present your chapter planning in the following JSON format, focusing on content and avoiding extraneous information.
Example format:
```
{{
"chapter1": {{"title": "<Chapter 1 Title>", "abstract": "<Detailed description of chapter content, including major events, character interactions, and plot developments. At least {chapter_abstract_num} words>"}},
"chapter2": {{"title": "<Chapter 2 Title>", "abstract": "<Detailed description of chapter content, including major events, character interactions, and plot developments. At least {chapter_abstract_num} words>"}},
...
}}
```
Only output the JSON result without any other information.
Premise: {premise}Synopsis: {synopsis}

B.2 Mid-Level Plot Structuring Template

CBN and *CEN* Generation prompt:

Prompt Template

The task involves generating a long story divided into multiple chapters. Each chapter has a title and main content provided. Your goal is to expand the main content of Chapter {chapter_no} to create a complete chapter. Additionally, you need to generate starting and ending plot node events (CBN and CEN) for this chapter.
To accomplish this, follow these steps:
1. Review the provided information for Chapter {chapter_no}, including the title and main content.
- For the current chapter, the title and main content are as follows:
‘‘‘
{chapter}‘‘‘
2. Expand the main content by adding additional details, descriptions, and dialogue to create a complete chapter.3. Ensure that the expanded content flows smoothly and maintains the tone and style of the story.4. Generate the starting plot node event (CBN) for this chapter. The CBN should be a subject-verb-object (SVO) triplet representing an event. Use the following format:
{{"subject": "<subject, the initiator of the action, a character or entity name>", "verb": "<verb, using the base form>", "object": "<object, the receiver of the action, a character or entity name, or an empty string if there’s no receiver>"
}}5. Make sure the CBN meets the plot node definition and structure requirements without including any extra information.
Generate the ending plot node event (CEN) for this chapter using the same format as the CBN.
6. Format the CBN and CEN in JSON format, ensuring they meet the plot node definition and structure requirements without any additional information.
7. Once you have completed the expansion of the main content and generated the CBN and CEN, proceed to the next chapter.
Please note that the story consists of multiple chapters, and you will need to repeat these steps for each chapter. You only need to output the final json output and nothing else.
Output Example format:
‘‘‘
{{
"CBN": {{"subject":"","verb":"","object":""}},
"CEN": {{"subject":"","verb":"","object":""}}
}}
‘‘‘
To assist you in this task, here are the titles and main content of previous and subsequent chapters:
Previous Chapters Content:
‘‘‘
{previous_chapter_abstracts}
‘‘‘
Subsequent Chapters Content:
‘‘‘
{subsquent_chapters_abstracts}‘‘‘

Pseudo CPN Generation prompt:

Prompt Template

I am currently generating a long story, divided into multiple chapters. In order to proceed with Chapter {chapter_no}, I need to create chapter plot nodes.
Plot Node Structure: Each plot node consists of a Subject-Verb-Object (SVO) tuple that represents an event. The format for each plot node is as follows: {{"subject":"<Initiator of the plot action, a character or entity name>","verb":"<Action, in verb base form>","object":"<Receiver of the plot action, a character or entity name, or an empty string if there's no receiver>"}}
You are an experienced fiction writer, specializing in crafting immersive narrative arcs. Your task is to create a new intermediate chapter plot node, to ensure a seamless and logical progression of the current chapter towards its chapter end node. Please follow these steps:
1. Understand and Analyze: First, thoroughly read and comprehend the existing chapter title, main content {chapter}, the beginning storyline node {CBN_data}, the concluding storyline node {CEN_data}, and the already developed intermediate nodes {CPNs}.
2. Strategic Planning: Consider how the story within the chapter should evolve to guide the plot smoothly from the current position towards the ending node.
3. Avoid Repetition: Be aware of the subsequent chapter's title and main content {next_chapter} to avoid overlapping or repeating narrative elements.
4. Focus on the Current Chapter: While crafting the storyline node, concentrate solely on the current chapter, avoiding the introduction of elements irrelevant to its context.
5. Please note that the current count of CPNs is {count}. The total number of CPNs should not exceed {chapter_CPN_maxnum}. Once the count exceeds {chapter_CPN_halfnum}, you should steer the middle plot nodes directly towards the concluding plot node to avoid clutter.
Your output should be a coherent and well-thought-out intermediate node that aligns with the chapter's overall theme and storytelling style.
Your output should be in JSON format, following the plot node structure and excluding any extraneous information.
Output format:
```
{{
"subject":"","verb":"","object":""
}}
```
You only need to output the final json output and nothing else.

Pseudo CPN Review prompt:

Prompt Template

We are currently working on generating a long story that is divided into multiple chapters. Each chapter has a title and main content. Your task is to expand on the main content of a specific chapter, referred to as chapter {chapter_no}.
To accomplish this, you need to create a plot node using a Subject-Verb-Object (SVO) triplet structure. The plot node should represent an event and follow this format:
{{
"subject": "<subject, the initiator of the plot action, a character or entity name>",
"verb": "<verb, using the base form of the verb>",
"object": "<object, the recipient of the plot action, a character or entity name, or an empty string if there's no recipient>"
}}
Before proceeding, here is some additional information:
- The current chapter's title and main content are {chapter}.
- The chapter's starting plot node is CBN:{CBN_data}.
- There are already some middle plot nodes generated, which are {CPNs}.
- We now need to assess whether the new middle plot node, {CPN_data}, meets certain requirements. If it does not, you should modify it and return the corrected middle plot node.
Here are the requirements for the new middle plot node:
1. The new middle node must enrich and be consistent with the main content of the chapter. It should also develop towards the Chapter End Node, CEN:{CEN_data}.
2. To prevent redundancy, avoid repeating plot nodes.
3. The most important principles you need to follow: The current number of middle nodes in the chapter is {count}, and the total number should not exceed {chapter_CPN_maxnum}. If the number of nodes exceeds {chapter_CPN_halfnum}, direct the plot towards the concluding node to avoid excessive and disorganized nodes.
4. When you think the storyline of the current chapter is complete, return {CEN_data} in order to achieve the purpose of stopping the generation of CPNs.
5. Provide related plot nodes information involving characters and objects in as follows. The information should include a time_stamp indicating the sequence of events. Ensure that the new plot node maintains logical and sequential consistency with the characters and objects.
Related nodes:
‘‘‘
{related_nodes}
‘‘‘
6. Inform the reason and basis for the modification in the ouput.
Please note that the output should be in JSON format, and the updated CPN should conform to the plot node definition and structure without any extraneous information.
‘‘‘
{{
"subject":"<Modified node subject>","verb":"<Modified node verb>","object":"<Modified node object>", "reson":"<Inform the reason and basis for the modification here>"
}}
‘‘‘
You only need to output the final json output and nothing else.

B.3 Detailed Story Writing Template

Text Block Generation prompt:

Prompt Template

We are in the process of generating a long story, divided into multiple chapters. For the current chapter, chapter number {chapter_no}, the title, main content, and plot points are as follows. You are required to generate the content of chapter {chapter_no} based on this information.
Plot Point Structure: A subject-verb-object (SVO) tuple representing an event, structured as:
‘‘‘
{{"subject":"<subject, the initiator of the plot action, either a character name or an entity>", "verb":"<verb, in its base form>","object":"<object, the receiver of the plot action, either a character name or an entity, empty string if none>"}}
‘‘‘
Current Chapter Title and Summary: {chapter}
Generation Requirements:
1. Generate the story based on the initial plot point (CBN): {CBN_data}, middle plot points: {CPNs}, and final plot point (CEN): {CEN_data}, as well as the chapter title and summary.
2. The story must strictly follow the provided plot points.
3. Include detailed emotional and psychological descriptions of characters, exploring their inner thoughts.
4. Incorporate rich environmental and background descriptions, with detailed literary and linguistic styles.
5. Clearly outline character development and the dynamics of relationships, addressing themes and symbolic elements.
6. Ensure plot coherence, encouraging complexity and creative elements.
7. Maintain consistency with the language style of previous chapters.
Previous Chapter Content:
‘‘‘
{generated_text}
‘‘‘
The generated content should be presented in JSON format, ensuring compliance with the plot point definitions and structure, without any extraneous information.
Output format:
‘‘‘
{{
"text":"<generated story content>"
}}
‘‘‘

B.4 Chat Template

For GPT-4o (OpenAI), LongWriter Models (Bai et al., 2024) and Suri-I-ORPO (Pham et al., 2024), we employ the following template for story generation, with particular emphasis on the length constraint: the story must exceed 3,000 words and be as long as possible. This focus on extended narrative generation aligns with the objective of our task, which centers on producing long-form stories.

Chat Template

You should write an engaging story based on the following requirements and writing prompt.

[Requirements]:
1. Feel free to use creativity to expand on the prompt and create an interesting and captivating narrative.
2. Ensure the story is at least 3,000 words long, but strive for more if the story naturally allows for it.
3. Create well-developed characters with distinct personalities, motivations, and backstories.
4. The plot should have multiple layers—include subplots or side characters that enrich the main storyline and add complexity to the story.
5. Balance the pacing of the narrative.
6. Use rich and evocative language that engages all the senses—sight, sound, smell, touch, and taste—to create a fully immersive reading experience.
7. Ensure the grammar, sentence structure, and overall coherence are polished, and the story flows smoothly from beginning to end.
8. Return only the final, complete generated story.

[Writing Prompts]:“‘{wp}“‘

B.5 Preference Evaluating Prompt Template

System Prompt

Please act as an impartial judge and evaluate the quality of the responses provided by two AI assistants to a user prompt. You will be given assistant A's answer (Story A) and assistant B's answer (Story B). Your job is to evaluate which assistant's story is better.
When evaluating the two stories, you should consider factors such as creativity, coherence, engagement, and relevance to the prompt. The story must be at least 3,000 words long, but we strongly encourage expanding the story beyond this minimum if the narrative allows for it. The more immersive and detailed, the better. Focus on comparing the strengths and weaknesses of each story. Highlight areas where one story excels over the other, such as storytelling techniques, narrative structure, character development, or emotional impact.
After providing your analysis and justification, you must output only one of the following choices as your final verdict with a label:
1. Assistant A is significantly better: [[A»B]]
2. Assistant A is slightly better: [[A>B]]
3. Tie, relatively the same: [[A=B]]
4. Assistant B is slightly better: [[B>A]]
5. Assistant B is significantly better: [[B»A]]
Example output: "My final verdict is tie: [[A=B]]".

User Prompt

<|User Prompt|>
{wp}
<|The Start of Assistant A's Story|>
{story_A}
<|The End of Assistant A's Story|>
<|The Start of Assistant B's Story|>
{story_B}
<|The End of Assistant B's Story|>