RAPID: Efficient Retrieval-Augmented Long Text Generation with Writing Planning and Information Discovery

Hongchao Gu[1], Dexun Li[2], Kuicai Dong[2], Hao Zhang[2], Hang Lv[1], Hao Wang[1*], Defu Lian[1], Yong Liu[2*], Enhong Chen[1*]
[1] University of Science and Technology of China, [2]Huawei Noah's Ark Lab
{hcgu, lvhang1001}@mail.ustc.edu.cn {wanghao3, liandefu, cheneh}@ustc.edu.cn
{lidexun, dong.kuicai, zhang.hao3, liu.yong6}@huawei.com

Abstract

Generating knowledge-intensive and comprehensive long texts, such as encyclopedia articles, remains significant challenges for Large Language Models. It requires not only the precise integration of facts but also the maintenance of thematic coherence throughout the article. Existing methods, such as direct generation and multi-agent discussion, often struggle with issues like hallucinations, topic incoherence, and significant latency. To address these challenges, we propose RAPID, an efficient retrieval-augmented long text generation framework. RAPID consists of three main modules: (1) Retrieval-augmented preliminary outline generation to reduce hallucinations, (2) Attribute-constrained search for efficient information discovery, (3) Plan-guided article generation for enhanced coherence. Extensive experiments on our newly compiled benchmark dataset, FreshWiki-2024, demonstrate that RAPID significantly outperforms state-of-the-art methods across a wide range of evaluation metrics (*e.g.,* long-text generation, outline quality, latency, etc). Our work provides a robust and efficient solution to the challenges of automated long-text generation.[1]

1 Introduction

Large Language Models (LLMs) have showcased significant proficiency in handling various natural language tasks (Chen et al., 2024; Hu et al., 2024; Wu et al., 2024), achieving near-human performance in tasks like summarizing lengthy documents (Liu et al., 2024b) and crafting poetry (Yu et al., 2024). Despite these achievements, generating long and knowledge-intensive texts (*e.g.,* encyclopedia articles) faces significant challenge (Shen et al., 2023). Such tasks demand not only the seamless integration of facts and narratives over extensive text but also the maintenance of thematic and stylistic consistency throughout the document.

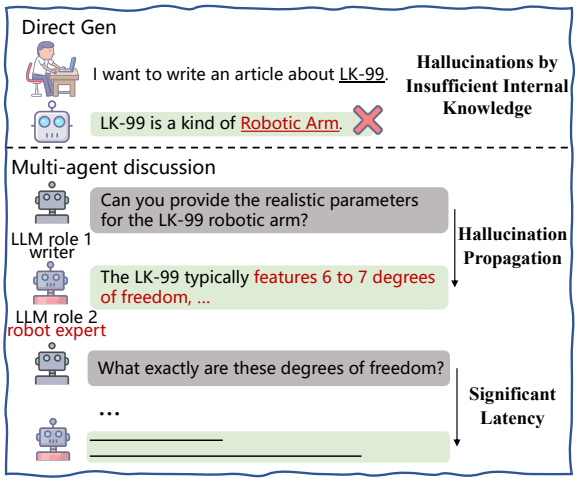

Figure 1: An example of generating a wiki-style article using various methods highlights distinct challenges and considerations. Direct generation may suffer from the large model's limited internal knowledge. While methods based on multi-agent discussions can provide broad coverage of the topic, they may also result in increased hallucinations and reduced efficiency.

Bai et al. (2024) focus on fine-tuning LLMs to generate long-text solely based on LLMs' internal knowledge. However, limited LLM's internal knowledge (You et al., 2023) can lead to significant hallucinations in its generation, as evident in Figure 1. Such phenomenon is further exacerbated when generating with fact-intensive content (Rawte et al., 2023). To address these issues, retrieval-augmented generation (RAG) techniques are commonly utilized to provide LLMs with additional retrieved content (Jiang et al., 2024b). Nonetheless, the effectiveness and scope of retrieved content can significantly influence the quality of generated long texts (Shao et al., 2024). Moreover, RAG-based long text generation relies on multiple rounds of retrieval, formulating dedicated search queries, and obtaining characteristic content for different sec-

[*]Corresponding authors.
[1]Code is publicly avaiblely at https://github.com/USTC-StarTeam/RaPID.

Findings of the Association for Computational Linguistics: ACL 2025, pages 16742–16763
July 27 - August 1, 2025

tions. This can cause thematic and stylistic inconsistency between sections (Zhou et al., 2023). A more effective approach is to mimic human writing practices (Rohman, 1965; Doyle and Center, 1994), directing LLMs to develop a structured outline prior to full-text generation.

A recent work, STORM (Shao et al., 2024) introduces a novel long-text generation strategy based on multi-agent discussions. This approach identifies multiple perspectives on each topic and uncovers information through interactions among agents assigned distinct roles, resulting in better performance compared to direct RAG-based methods.

Despite its promising results, long text generation based on multi-agent disscussion faces three challenges as follows: **(1)** *writing intentions are often general and ambiguous* , but agents are constrained by the internal knowledge of LLMs, making it highly possible to misinterpret the writing intention from the very beginning and generate hallucinations. **(2)** *agent interactions lack self-correction mechanisms*, leading to a failure to detect hallucinations and allowing ineffective discussions to perpetuate errors unchecked. **(3)** *long texts inherently involve intricate long-range dependencies and complex logical structures*. Consequently, maintaining consistency and accuracy throughout the generation process remains challenging.

To address these challenges, we propose RAPID, an efficient **R**etrieval-**A**ugmented long text generation framework with writing **P**lanning and **I**nformation **D**iscovery. Specifically, **(1)** we design a retrieval-augmented outline generation module. We first establish an outline corpus containing approximately 2.6 million example outlines from Wikipedia, covering a diverse range of topics and structures. Given a question, it is disambiguated via results from web search, and the refined intent is used to retrieve outline examples from established outline corpus. These outline examples are used as context for high-quality outline generation. **(2)** To comprehensively gather information across multiple topics, we extract attributes from the outline. We maintain a comprehensive attribute buffer and convert the extracted attributes into search queries for attribute-constrained search. The collected information contributes to iteratively updating the outline and serves as reference material to support the generation of the final long text. **(3)** Most importantly, we design a novel structured writing plan. It is derived from the outline, to guide full article generation. Specifically, the writing plan is a topological graph that plots section dependencies and writing sequences.

Our contributions are summarized as follows:

- We revisit the automated knowledge-intensive long text generation, particularly focusing on the unified consideration of pre-writing and generation stages for generating wiki-style articles.
- We propose RAPID, a framework that leverages retrieval-augmented outline generation, efficient information discovery, and logic writing plan guidance to generate comprehensive and knowledge-intensive articles.
- We construct a new dataset, and extensive experiments demonstrate the effectiveness of RAPID in terms of factuality and coherence.

2 Related Works

Long-form Text Generation Long-form text generation (Tan et al., 2021; Guan et al., 2021; Yang et al., 2022) has been a significant focus in NLP research, even prior to the emergence of large language models. Recently, researchers have achieved considerable success in various applications of long text generation using LLMs, including creative writing (Lei et al., 2024; Li et al., 2024; Yang et al., 2023), scientific survey (Kang and Xiong, 2024; Wang et al., 2024) and expository writing (Balepur et al., 2023). Previous efforts to enhance long text generation have often involved improving models' abilities to produce extended outputs by constructing crafted SFT dataset (Xu et al., 2024). This paper, however, primarily focuses on generating knowledge-intensive long articles based on topic retrieval. For instance, Shao et al. (2024) proposed a pre-writing method that utilizes perspective-guided questioning and outline generation, which automates wiki-style article generation from scratch. Building on this foundation, Jiang et al. (2024a) introduced a multi-agent dialogue and user interaction mechanism, which further enhances the large language model's ability to explore unknown unknowns. Additionally, Pham et al. (2024) and Bai et al. (2024) proposed to enhance the ability of LLMs to directly generate long outputs by improving LLM alignment methods. However, these methods still face challenges related to hallucinations, efficiency, and consistency, which are the primary focus of optimization in this paper.

Retrieval Augmented Generation Retrieval-Augmented Generation (Lewis et al., 2020) represents an innovative approach that integrates the advantages of information retrieval with language generation. The performance of RAG heavily depends on the accuracy of the retrieved information (Mallen et al., 2023; Dong et al., 2025a). Some works improve retrieval performance through adaptive retrieval (Asai et al., 2023; Liu et al., 2024a; Li et al., 2025) or query rewriting (Gao et al., 2023; Ma et al., 2023). More recently, Edge et al. (2024) proposes to enhance the accuracy of RAG by automatically building knowledge graphs. Although these methods have demonstrated significant effectiveness in short-format output tasks (Joshi et al., 2017; Yang et al., 2018), they remain inadequate for application in knowledge-intensive long text generation tasks, which usually require multi-faceted information retrieval and fine-grained information filtering.

3 Methods

In this section, we introduce RAPID, an efficient and effective framework for automated wiki-style article generation. We first formulate the article generation problem in Section 3.1. Next, we detail the retrieval-augmented outline generation process in Section 3.2, the attribute-constrained search in Section 3.3, and plan-guided long-text generation in Section 3.4. The overall framework of RAPID is illustrated in Figure 2.

3.1 Problem Formulation

The goal of this task is to automatically generate a comprehensive wiki-style article, denoted as $\mathcal{D}$, based on a given topic t. This process involves generating a clear outline and crafting coherent sections that thoroughly explore the topic. Formally, the task includes three main steps: (i) Initial outline generation: given a topic t, we first generate an initial outline $\mathcal{O}_{ini}$ related to t, (ii) Information gathering and update: we then use the search engine to collect a diverse set of information from the knowledge source $\mathcal{S}$ as the reference $\mathcal{R}$ and iteratively refine $\mathcal{O}_{ini}$ to produce the final outline $\mathcal{O}$; (iii) Article crafting: finally, the complete article $\mathcal{D}$ is generated by elaborating on the final outline $\mathcal{O}$ using the reference $\mathcal{R}$.

3.2 Retrieval Augmented Outline Generation

"Well begun is half done." A clear and logically coherent outline is crucial for the writing process, which serves as a roadmap to guide the detailed content unfold in appropriate places and to ensure overall cohesion (Fan and Gardent, 2022). However, existing methods rely on generating an initial outline using the model's internal knowledge, which will inevitably introduce inaccuracies or misconceptions. This outline is then iteratively refined through RAG. However, such inaccuracies may escalate as the writing progresses, leading to topic deviations. To address this, we employ retrieval-augmented generation during the initial outline generation phase to improve writing intent recognition and reduce hallucinations. Additionally, we retrieve similar topic outline examples from a curated corpus, generating a high-quality preliminary outline that is further refined through an efficient information retrieval module.

Intent recognition. To reduce ambiguity, we begin by using an intention recognition module to clarify intentions and maintain focus. Specifically, for a given topic t, we first query a search engine for relevant information and then generate an initial summary of the results. This process is formalized as follows:

$$\mathcal{S}_t = \texttt{Sch}(t, \mathcal{S}) \text{ and } I_{\text{brief}} = \texttt{LLM}(t, \mathcal{S}_t) \quad (1)$$

where I_{brief} represents the brief summary of the topic, and $\mathcal{S}_t$ denotes the search results obtained by the search engine Sch from the knowledge base $\mathcal{S}$.

Outline example retrieval. To improve the quality of the generated initial outline, we retrieve high-quality outline examples from a curated corpus based on the given topic, which serve as few-shot examples for outline generation. Specifically, to construct the corpus $\mathcal{C}$, we extract outlines from the extensively crawled structured articles in the Wiki archives[2]. We then concatenate the topic t and its brief summary I_{brief} to facilitate query expansion, and use a dense retriever $\texttt{Ret}_1$ to retrieve similar topics from the constructed Wiki corpus $\mathcal{C}$. We denote these retrieved topics as $\mathbb{T} = \{t'_1, t'_2, ..., t'_n\}$, with their corresponding outlines $\mathbb{O} = \{\mathcal{O}'_1, \mathcal{O}'_2, ..., \mathcal{O}'_n\}$. We have:

$$(\mathbb{T}, \mathbb{O}) = \texttt{Ret}_1(t, I_{\text{brief}}, \mathcal{C}) \quad (2)$$

After retrieval, we combine the topic t, summary I_{brief} as the reference context, and the retrieved outlines $\mathbb{T}$ and $\mathbb{O}$ as few-shot examples to instruct

[2] https://dumps.wikimedia.org/

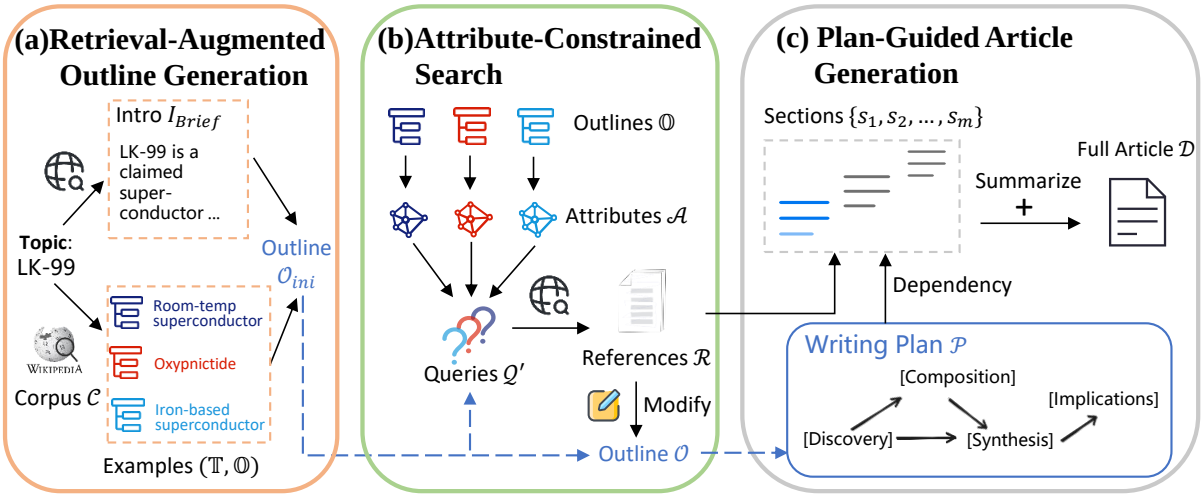

Figure 2: The framework of RAPID, which consists of three main stages: (a) Retrieval-Augmented Outline Generation, where an initial outline is created based on a brief introduction and examples; (b) Attribute-Constrained Search, which leverages an attribute-based mechanism to discover relevant information and refine the outline accordingly; and (c) Plan-Guided Article Generation, where a structured writing plan is developed based on dependencies between sections, resulting in a more coherent and fluent article. The blue dashed lines illustrate how the outline evolves throughout the processes of information discovery and writing planning.

LLM to generate the initial outline, denoted as $\mathcal{O}_{ini}$:

$$\mathcal{O}_{ini} = \texttt{LLM}(t, \mathcal{S}_t, \mathbb{T}, \mathbb{O}) \tag{3}$$

3.3 Attribute-Constrained Search

After generating the initial outline $\mathcal{O}_{ini}$, we design an attribute-constrained information collection module to enhance the efficiency and comprehensiveness of information gathering, further refining the $\mathcal{O}_{ini}$ to produce the final outline $\mathcal{O}$.

Attribute & Query Generation. In the process of generating long-form text, it is crucial to ensure that the content is comprehensive and relevant. To this end, we introduce the concept of attributes. Each attribute is defined as a distinct, indivisible concept to prevent overly complex queries, thereby effectively summarizing the essential information for writing the full article. By breaking down the outline into distinct attributes, we can focus on capturing the most pertinent details without introducing overwhelming complexity. We prompt LLM to extract the possible attribute from the current outline $\mathcal{O}_{ini}$ and maintain an attribute buffer, $\mathcal{A}$. Such buffer encompasses the key aspects necessary for composing the final article. We then prompt LLM to transform the attributes into queries, $\mathcal{Q}$, that are related to the article's title and are suitable for search engine [3]. The collected information will be added into the reference set $\mathcal{R}$. We have:

$$\mathcal{A} = \texttt{LLM}(\mathcal{O}_{ini}), \mathcal{Q} = \texttt{LLM}(\mathcal{A}) \text{ and } \mathcal{R} = \texttt{Sch}(\mathcal{Q}, \mathcal{S}) \tag{4}$$

We have fully parallelized the search process to enhance the efficiency. The example queries are then selectively merged into a set of queries $\mathcal{Q}$ by a LLM. The merged queries $\mathcal{Q}'$ are used in parallel with search engines to gather information, which serves as references $\mathcal{R}$ for the outline refinement and subsequent writing process. This attribute-constrained approach is highly parallelized, enabling effective and efficient information gathering.

Outline Refinement To further align the initial outline $\mathcal{O}_{ini}$ with the desired structure, we provide the LLMs with the titles of all references from $\mathcal{R}$ and instruct LLMs to generate modifications to $\mathcal{O}_{ini}$. These modifications can include actions such as `[add section]`, `[delete section]`, or `[do nothing]`, ensuring that the outline aligns with the most current information and insights available.

We will iterate the outline refinement process until the attribute buffer $\mathcal{A}$ and outline converge,

[3]Note that such attributes can also be directly provided or modified by humans to guide relevant retrieval during long-form text generation, incorporating points that are of interest or that they wish to highlight.

or the search limits are reached. Thus, the final outline $\mathcal{O}$ is generated based on $\mathcal{O}_{ini}$ and $\mathcal{R}$:

$$\mathcal{O} = \texttt{LLM}(\mathcal{O}_{ini}, \mathcal{R}) \tag{5}$$

3.4 Plan-Guided Article Generation

Plan Generation Existing methods directly generate the article based on outline $\mathcal{O}$, such a simple one-way writing logic may cause the content to gradually deviate from the initial section. A long article has a relatively complex structure (Dong et al., 2024), with interdependencies between sections. To maintain the overall coherence and consistency, we introduce an additional writing plan, denoted as $\mathcal{P}$, which provides a carefully planned writing order derived from the outline $\mathcal{O}$. Specifically, we first extract all the first-level section titles from $\mathcal{O}$, denoted as $\{s_1, s_2, \ldots, s_m\}$. We then define the "dependency" of a section as the content from other sections that must be referenced before developing that particular section, ensuring a logical flow and integration throughout the article.

Leveraging the planning capabilities of LLMs, we construct plan $\mathcal{P}$ by providing human-written few-shot examples. Specifically, the plan $\mathcal{P}$ consists of a list that defines the dependencies of each section, with "None" indicating no dependencies on other sections. We then calculate its topological order to ensure the feasibility of the plan, and the full article is generated section by section according to this order. In the rare instances where the generation of $\mathcal{P}$ fails, we will revert to the more conventional approach of generating each section in parallel.

The generation of the particular section s_i can be formulated as:

$$\begin{aligned} \mathcal{P} &= \texttt{LLM}(\{s_1, s_2, \ldots, s_m\}), \\ \mathcal{R}_{s_i} &= \texttt{Ret}_1(s_i, \mathcal{P}, \mathcal{R}, \mathcal{O}), \\ s_i &= \texttt{LLM}(t, \mathcal{P}, \mathcal{O}, \mathcal{R}_{s_i}). \end{aligned} \tag{6}$$

where $\mathcal{R}_{s_i}$ represents the references retrieved from $\mathcal{R}$ that are needed to generate s_i.

After generating all the sections, we follow established methodologies by concatenating them to create a comprehensive summary of the topic. This process culminates in the creation of a complete wiki-style article.

4 Experiments

4.1 Dataset

To evaluate the capabilities of RAPID, we introduce the FreshWiki-2024 dataset, a chronological extension of the original FreshWiki dataset (Shao et al., 2024) which consists of human-authored Wikipedia articles covering 100 distinct topics. To address temporal data contamination concerns, FreshWiki-2024 follows the rigorous curation protocol of its predecessor and only includes Wikipedia entries revised during 2024, ensuring all evaluation materials postdate conventional model training cutoffs. For robust performance evaluation, we randomly selected 100 representative topics from FreshWiki-2024 to evaluate both RAPID and baseline methods. Comprehensive dataset statistics and details are provided in Appendix A.

4.2 Baselines

We compared RAPID with several different LLM-based methods:

- **RAG**: It generates an outline or full article based on the information retrieved from a search related to the writing topic.
- **Outline-driven RAG (oRAG)**: Building upon RAG, this method first generates an outline. It then uses the section titles to retrieve additional relevant information and generates the full article, section by section, using the retrieved content.
- **STORM** (Shao et al., 2024): A writing system designed to generate wiki-style articles from scratch. It majors in the pre-writing stage by generating the outline through retrieval and perspective-guided question-asking.
- **Co-STORM** (Jiang et al., 2024a): An enhanced version of STORM that incorporates user interaction and multi-agent discussion, further improving the exploration of previously unknown aspects within the topic.

4.3 Evaluation Metrics

Outline Quality. To objectively assess outline generation quality, we propose an evaluation framework based on section title alignment. Rather than employing NER-based metrics as in STORM (refer to Appendix B.1 for its limitations), we perform direct string matching between generated section titles and human-authored ground truth titles. We calculate recall, precision, and F1 score, providing a balanced assessment of outline comprehensiveness and conciseness.

Article Quality. Evaluating long-form text generation remains a challenging task. To comprehensively assess the quality of generated articles, we consider four key aspects: (1) **Similarity**: We use

	Method	LLM Evaluation				Similarity			Factuality			Info
		Intere.	Organiz.	Relev.	Cover.	Rouge-1	Rouge-L	Entity Recall	Precision	Claims	F1@300	**Diversity**
Qwen-Max	RAG	3.66	4.24	4.56	**4.20**	33.22	14.72	11.23	46.50	193.90	54.09	0.456
	oRAG	3.32	3.79	3.87	3.61	40.04	14.49	9.27	43.90	396.30	65.90	0.552
	STORM	3.90	4.44	4.49	4.15	41.35	15.07	10.85	44.30	440.90	68.08	0.618
	Co-STORM	3.92	4.16	4.51	4.10	32.43	13.08	8.38	49.00	230.90	59.88	0.580
	Rapid	**3.96**	**4.49**	**4.69**	4.13	**44.65**	**15.86**	**12.60**	**51.10**	**493.70**	**77.98**	**0.650**
	w/o plan	3.85	**4.49**	4.58	4.06	44.06	15.62	12.45	46.80	464.6	71.88	–
DeepSeek-v3	RAG	4.03	4.39	4.66	4.05	25.27	12.23	9.40	**57.50**	103.10	43.02	0.469
	oRAG	3.68	4.23	4.70	4.11	24.74	10.74	6.53	52.40	125.40	46.50	0.498
	STORM	4.23	**4.58**	4.80	4.14	40.05	14.78	**11.81**	48.00	**395.50**	70.38	0.607
	Co-STORM	3.88	3.89	4.10	3.81	23.09	9.68	5.87	52.10	174.50	54.97	0.591
	Rapid	**4.37**	4.53	**4.85**	**4.21**	**40.64**	**15.16**	10.94	54.70	337.70	**73.62**	**0.670**
	w/o plan	4.27	4.52	4.72	4.13	40.08	14.87	10.86	53.80	311.3	70.86	–
GPT-4o	RAG	4.18	4.42	**4.86**	4.02	30.02	12.98	10.80	**56.80**	128.60	48.86	0.429
	oRAG	3.46	4.19	4.50	4.07	31.01	12.18	7.85	46.20	261.00	60.35	0.539
	STORM	4.16	4.46	4.61	4.09	40.56	14.46	**12.19**	45.60	398.10	67.88	0.626
	Co-STORM	4.31	4.30	4.78	4.14	34.14	13.07	8.83	44.80	276.80	60.31	0.584
	Rapid	**4.38**	**4.61**	4.73	**4.20**	**43.35**	**15.33**	12.01	48.80	**448.10**	**73.57**	**0.650**
	w/o plan	4.35	4.47	4.73	4.16	39.19	14.35	10.65	45.00	364.70	65.69	–

Table 1: Results of article quality evaluation. The best results of each metric are marked in **bold**. **LLM Evaluation** uses an LLM to assess the generated articles across four different dimensions. The LLM Evaluation uses a 1-5 scale. **Similarity** assesses the resemblance between generated and real articles. **Factuality** evaluates the quantity and accuracy of facts in the generated content. **Info Diversity** evaluates the diversity of the searched information. *w/o plan* denotes direct parallel generation without a writing plan. Since search components remain unchanged in this scenario, "Info Diversity" is not applicable and denoted by "–".

Rouge (Lin, 2004) scores and entity recall from a FLAIR NER model (Akbik et al., 2019) to compare the similarity between generated articles and human-written articles. (2) **Factuality**: We also assess the factual accuracy of the generated articles. Specifically, we employ FActScore (Min et al., 2023), an automated long-text factuality evaluation framework based on LLMs. We report the precision and the number of claims of FActScore. Additionally, we include F1@300 as defined in FactAlign (Huang and Chen, 2024), to penalize articles that contain few factual statements. (3) **LLM Evaluation**: To evaluate the quality of generated articles at the article level, we utilized `Prometheus-7b-v2.0` (Kim et al., 2024), an open-source model that supports a 5-point rubric for text evaluation. We employed the same version of the criteria used by STORM, focusing on four aspects: *Interest*, *Organization*, *Relevance*, and *Coverage*. (4) **Diversity**: We measure the diversity of collected information using the *info diversity* metric proposed by CoSTORM, which quantifies the variety of information included in the generated articles. Further details on the evaluation metrics can be found in Appendix B.2.

4.4 Implementation Details

We implemented RAPID using the DSPy framework[4], with all prompts detailed in Appendix C. Three foundation models were employed as backbones: `gpt-4o-2024-11-20`, `qwen-max-2024-09-19`, and `deepseek-v3`. For web search functionality across all methods, we integrated the Google Custom Search API[5], retrieving the top-5 most relevant results per query to balance precision and computational overhead. To prevent data leakage, we explicitly excluded official Wikipedia pages related to target topics during web search and outline retrieval. We implemented $\texttt{Ret}_1$ and $\texttt{Ret}_2$ using `e5-large-v2` (Wang et al., 2022) and `paraphrase-MiniLM-L6-v2` (Reimers and Gurevych, 2019), respectively. The Wikipedia dump[6] (version 2024-08-01) served as the retrieval corpus for outline generation.

5 Results and Analysis

5.1 Main Results

Table 1 presents the article quality evaluation of RAPID across different foundation models. Our

[4] https://github.com/stanfordnlp/dspy

[5] https://developers.google.com/custom-search/v1/overview

[6] https://dumps.wikimedia.org/backup-index.html

	Method	Recall	Precision	F_1
QwenMax	RAG/oRAG	10.53	**9.05**	5.48
	STORM	13.86	6.23	4.94
	Co-STORM	3.8	8.68	4.93
	RAPID	**19.83**	8.50	**10.86**
DS-v3	RAG/oRAG	9.97	5.79	6.82
	STORM	11.95	4.90	6.56
	Co-STORM	5.13	8.78	5.82
	RAPID	**20.82**	**14.25**	**16.07**
GPT-4o	RAG/oRAG	10.28	5.10	6.28
	STORM	11.12	3.66	5.19
	Co-STORM	2.83	4.09	2.93
	RAPID	**22.77**	**15.92**	**17.52**

Table 2: Results of outline quality evaluation. We assess the resemblance between generated and real outline.

approach demonstrates substantial improvements over baseline methods in multiple dimensions. In terms of LLM-based evaluation and similarity metrics, RAPID outperforms existing baselines on most criteria while remaining competitive on the remaining indicators. The factuality assessment reveals that RAPID achieves significant improvements in F1@300 scores across all foundation models compared to baselines, indicating its superior capability to maintain factual accuracy while preserving high information density. Furthermore, RAPID attains the highest info diversity scores among all tested configurations, suggesting that our method effectively balances information discovery efficiency with diversity preservation.

Additionally, we observed that certain simple baseline methods achieved unexpectedly high scores on specific metrics. For instance, RAG exhibits an unusually high relevance score on GPT-4o. This phenomenon may be attributed to the lower information density of its generated content, as evidenced by the fact that the number of claims it produces is less than one-third of that generated by RAPID. This observation underscores the challenge of evaluating long-form content quality using a single metric. In contrast, RAPID demonstrates superior performance across multiple evaluation criteria, reinforcing its robustness and effectiveness.

The quality evaluation results of outlines are presented in Table 2. RAPID significantly outperforms other methods across different foundation models, particularly in recall and F1-score metrics. This demonstrates its superior ability to maintain thematic focus while achieving comprehensive coverage of research topics. Notably, although

Backbone	Avg. Nodes	Avg. Edges	Dependency Density	Longest Path
Qwen-Max	8.72	16.64	2.09	2.79
DeepSeek-v3	7.21	12.18	1.91	2.62
GPT-4o	8.43	11.86	1.55	3.59

Table 3: Results of the graph metrics of Writing Plan.

STORM achieves relatively higher recall compared to other baselines, RAPID can efficiently generate higher-quality outlines through retrieval of similar topics, without requiring complex multi-turn agent discussion mechanisms.

5.2 Ablation Studies

To evaluate the impact of the writing plan, we conducted an ablation study by removing the Plan-Guided Article Generation module and instead employing a parallel section-wise generation approach (denoted as "w/o plan"). As shown in Table 1, the absence of a writing plan led to performance degradation across multiple metrics. Without contextual paragraph awareness, the model exhibited stylistic inconsistencies and coherence issues between sections, explaining the decline in organization scores. Additionally, generating sections in isolation resulted in redundant content, as each paragraph attempted to cover multiple aspects of the topic independently. While techniques such as STORM can partially mitigate these issues through post hoc refinement, maintaining coherence and quality in long-form text remains challenging due to inherent output length limitations. In contrast, our structured planning approach explicitly defines intersectional dependencies, enhancing overall coherence and enabling the generation of longer, more cohesive articles. These findings underscore the critical role of planning in long text generation.

5.3 Writing Plan Analysis

Our method relies on constructing a directed acyclic graph (DAG) of sections prior to writing, which defines the contextual dependencies between sections. In this section, we analyze the Writing Plan's functionality through graph-theoretical metrics. By representing sections as nodes and intersectional dependencies as directed edges, we formally define the Writing Plan as a DAG structure $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ denotes the set of n nodes (sections), and $E \subseteq V \times V$ represents the directed edges indicating dependen-

cies. To quantify the "dependency density" of the Writing Plan, we define it as the ratio of the number of edges to the maximum possible edges in a minimally connected DAG:

$$D(G) = \frac{|E|}{n-1}, \tag{7}$$

where $|E|$ is the cardinality of the edge set E, and $n-1$ corresponds to the maximum number of edges in a linear chain structure. For example, in a sliding window of length $k = 1$, where each section depends only on its immediate predecessor, the number of edges is $|E| = n - 1$, yielding a dependency density of $D(G) = 1$. Besides dependency density, we also report key metrics to describe the DAG structure: average nodes, edges, and longest path length. The longest path highlights the critical dependency sequence, determining the minimum steps to traverse the Writing Plan.

The results for the plans generated by three backbone models are summarized in Table 3. These results reveal that while the average number of sections across the plans remains relatively consistent, the number of dependencies exhibits more significant variation. Notably, the dependency density for all three backbones exceeds 1, ranging from 1.55 to 2.09. This indicates that the generated plans effectively transcend simple linear structures, capturing more complex long-range dependencies. These findings demonstrate the strong generalization capability of our method. The detailed case study of the writing plan is provided in Appendix E.

Furthermore, an excessively long critical path can lead to reduced parallelism during section generation, thereby decreasing efficiency. However, the average longest path length of the plans generated by the three backbones does not exceed 4, ensuring that the generation time of RAPID remains within an acceptable range. Further evidence can be found in the detailed efficiency analysis provided in Appendix D.

6 Human Evaluation

To gain deeper insights into the quality of the generated articles, we recruited 10 master-degree level volunteers to conduct a human evaluation. We randomly selected 20 topics and requested the volunteers to evaluate articles generated by both RAPID and STORM for each topic. Each article was assigned to two different volunteers to ensure a balanced and unbiased assessment. We provided the volunteers with the same evaluation criteria as those used in the LLM-based assessment (see Table 6), which assess four dimensions: Interest Level, Organization, Relevance, and Coverage. Volunteers rated each dimension on a scale from 1 to 5. Additionally, they were asked to indicate their preference for each pair of articles. To mitigate potential bias, the order of methods in each article pair was randomized.

Method	Intere.	Organiz.	Relev.	Cover.	#Prefer
STORM	**4.03**	3.80	3.75	4.05	13
RAPID	3.98	**4.03**	**4.25**	**4.15**	**27**

Table 4: Results of human evaluation. We had human evaluators evaluate the articles across four dimensions using a fine-grained scale ranging from 1 to 5, and we documented their preferences accordingly.

As shown in Table 3, our RAPID significantly outperforms STORM in terms of Organization, Relevance, and Coverage. Additionally, it achieves a notable improvement in overall preference, despite showing a slight disadvantage in Interest. These results indicate that RAPID generates articles of higher overall quality compared to STORM.

Notably, the significantly lower Relevance score of STORM may be attributed to hallucination propagation caused by its agent-based discussion mechanism. For example, feedback from volunteers revealed that when processing the topic Vultures 1 (album), STORM's pre-writing phase—relying solely on LLM-driven dialogue for information collection—misinterpreted the task, mistakenly assuming it involved generating content about vultures (the animal). This misunderstanding led to irrelevant, animal-related perspectives, rendering the entire information collection and writing process ineffective. In contrast, RAPID addresses this issue by employing retrieval-based methods to ensure accurate intent recognition from the outset. This approach enables a more precise and efficient information collection process, significantly reducing the risk of misinterpretation and ultimately improving the quality of the generated content.

7 Conclusion

In this paper, we addressed the challenge of knowledge-intensive long-text generation, with a particular focus on encyclopedia-style article generation. We introduced RAPID, an efficient and effective retrieval-augmented writing system that

integrates a structured writing plan and information discovery mechanisms. Our extensive experiments and human evaluations demonstrate that our model outperforms state-of-the-art baselines in both effectiveness and efficiency. Additionally, our model maintains scalability for other knowledge-intensive long-text writing tasks. In the future, we plan to explore the potential of LLMs in other domain-specific writing tasks, such as financial reports.

Limitations

Our proposed RAPID, while demonstrating efficacy in long-form Wikipedia article generation, is subject to two key constraints that warrant discussion. The current methodology exclusively focuses on textual content generation, overlooking the critical integration of multimodal elements such as images and tabular data that significantly enhance article credibility and reader engagement in authentic Wikipedia entries. Furthermore, the evaluation framework remains confined to the Wikipedia domain, leaving untested the method's adaptability to other long-form generation scenarios like financial reporting or academic writing, which demand distinct structural precision and factual rigor. We plan to use more advanced data augmentation strategies (Yin et al., 2024) in the future to improve the model's performance on texts of specific styles or domains. These limitations underscore the need for subsequent research to expand multimodal integration capabilities and conduct cross-domain validation, thereby enhancing both the practical utility and generalizability of the proposed RAPID.

Ethics Statement

Our research focuses on automated long-text generation using LLMs, specifically for producing Wikipedia-style articles. While this technology holds great potential for enhancing knowledge accessibility and streamlining content creation, it also presents several ethical challenges that we acknowledge and actively address.

One key concern is that the generated content may delve into sensitive topics, as the model synthesizes information from diverse sources. Despite our extensive efforts to exclude controversial subjects during dataset construction, there remains a risk of producing biased, misleading, or inappropriate content. Such outputs could inadvertently activate LLM content safety mechanisms. We take this issue seriously and underscore the importance of advancing content filtering techniques and bias mitigation strategies to ensure the responsible use of this technology.

Another significant challenge lies in ensuring factual accuracy. Despite implementing measures to minimize errors, the model may still generate incorrect or hallucinated information. Worse still, if misused, this technology could actively contribute to the spread of false information, amplifying its potential societal impact. Addressing this issue calls for future research aimed at enhancing factual consistency, developing safeguards against misuse, and integrating more robust verification systems to validate the generated content.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (62441239, U23A20319, 62441227, 62472394, 62202443) as well as the Anhui Province Science and Technology Innovation Project (202423k09020011), Anhui Provincial Science and Technology Major Project (2023z020006).

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*.

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 [cs].

Nishant Balepur, Jie Huang, and Kevin Chang. 2023. Expository text generation: Imitate, retrieve, paraphrase. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 11896–11919, Singapore. Association for Computational Linguistics.

Guizhen Chen, Liying Cheng, Anh Tuan Luu, and Lidong Bing. 2024. Exploring the potential of large language models in computational argumentation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2309–2330, Bangkok, Thailand. Association for Computational Linguistics.

Kuicai Dong, Yujing Chang, Xin Deik Goh, Dexun Li, Ruiming Tang, and Yong Liu. 2025a. Mmdocir: Benchmarking multi-modal retrieval for long documents. *Preprint*, arXiv:2501.08828.

Kuicai Dong, Yujing Chang, Shijie Huang, Yasheng Wang, Ruiming Tang, and Yong Liu. 2025b. Benchmarking retrieval-augmented multimomal generation for document question answering. *Preprint*, arXiv:2505.16470.

Kuicai Dong, Derrick Goh Xin Deik, Yi Quan Lee, Hao Zhang, Xiangyang Li, Cong Zhang, and Yong Liu. 2024. MC-indexing: Effective long document retrieval via multi-view content-aware indexing. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2673–2691, Miami, Florida, USA. Association for Computational Linguistics.

C.S. Doyle and Educational Resources Information Center. 1994. *Information Literacy in an Information Society: A Concept for the Information Age*. ERIC reports. DIANE Publishing.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *Preprint*, arXiv:2404.16130.

Angela Fan and Claire Gardent. 2022. Generating biographies on Wikipedia: The impact of gender bias on the retrieval-based generation of women biographies. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8561–8576, Dublin, Ireland. Association for Computational Linguistics.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777, Toronto, Canada. Association for Computational Linguistics.

Jian Guan, Xiaoxi Mao, Changjie Fan, Zitao Liu, Wenbiao Ding, and Minlie Huang. 2021. Long text generation by modeling sentence-level and discourse-level coherence. 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 6379–6393, Online. Association for Computational Linguistics.

Yuchen Hu, Chen Chen, Chao-Han Yang, Ruizhe Li, Dong Zhang, Zhehuai Chen, and EngSiong Chng. 2024. GenTranslate: Large language models are generative multilingual speech and machine translators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 74–90, Bangkok, Thailand. Association for Computational Linguistics.

Chao-Wei Huang and Yun-Nung Chen. 2024. Factalign: Long-form factuality alignment of large language models. *Preprint*, arXiv:2410.01691.

Yucheng Jiang, Yijia Shao, Dekun Ma, Sina J. Semnani, and Monica S. Lam. 2024a. Into the unknown unknowns: Engaged human learning through participation in language model agent conversations. *Preprint*, arXiv:2408.15232.

Ziyan Jiang, Xueguang Ma, and Wenhu Chen. 2024b. Longrag: Enhancing retrieval-augmented generation with long-context llms. *Preprint*, arXiv:2406.15319.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Hao Kang and Chenyan Xiong. 2024. ResearchArena: Benchmarking LLMs' Ability to Collect and Organize Information as Research Agents. *arXiv preprint*. ArXiv:2406.10291 [cs].

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024. Prometheus 2: An open source language model specialized in evaluating other language models. *Preprint*, arXiv:2405.01535.

Huang Lei, Jiaming Guo, Guanhua He, Xishan Zhang, Rui Zhang, Shaohui Peng, Shaoli Liu, and Tianshi Chen. 2024. Ex3: Automatic Novel Writing by Extracting, Excelsior and Expanding. *arXiv preprint*. ArXiv:2408.08506 [cs].

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Wenjun Li, Dexun Li, Kuicai Dong, Cong Zhang, Hao Zhang, Weiwen Liu, Yasheng Wang, Ruiming Tang, and Yong Liu. 2025. Adaptive tool use in large language models with meta-cognition trigger. *Preprint*, arXiv:2502.12961.

Yunzhe Li, Qian Chen, Weixiang Yan, Wen Wang, Qinglin Zhang, and Hari Sundaram. 2024. Advancing precise outline-conditioned text generation with task duality and explicit outline control. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2362–2377, St. Julian's, Malta. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Huanshuo Liu, Hao Zhang, Zhijiang Guo, Jing Wang, Kuicai Dong, Xiangyang Li, Yi Quan Lee, Cong Zhang, and Yong Liu. 2024a. Ctrla: Adaptive retrieval-augmented generation via inherent control. *Preprint*, arXiv:2405.18727.

Yixin Liu, Kejian Shi, Katherine He, Longtian Ye, Alexander Fabbri, Pengfei Liu, Dragomir Radev, and Arman Cohan. 2024b. On learning to summarize with large language models as references. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8647–8664, Mexico City, Mexico. Association for Computational Linguistics.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore. Association for Computational Linguistics.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, Singapore. Association for Computational Linguistics.

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

Vipula Rawte, Swagata Chakraborty, Agnibh Pathak, Anubhav Sarkar, S.M Towhidul Islam Tonmoy, Aman Chadha, Amit Sheth, and Amitava Das. 2023. The troubling emergence of hallucination in large language models - an extensive definition, quantification, and prescriptive remediations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2541–2573, Singapore. 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*. Association for Computational Linguistics.

D. Gordon Rohman. 1965. Pre-writing the stage of discovery in the writing process. *College Composition and Communication*, (2):106–112.

Yijia Shao, Yucheng Jiang, Theodore Kanell, Peter Xu, Omar Khattab, and Monica Lam. 2024. Assisting in writing Wikipedia-like articles from scratch with large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6252–6278, Mexico City, Mexico. Association for Computational Linguistics.

Zejiang Shen, Tal August, Pao Siangliulue, Kyle Lo, Jonathan Bragg, Jeff Hammerbacher, Doug Downey, Joseph Chee Chang, and David Sontag. 2023. Beyond summarization: Designing ai support for real-world expository writing tasks. *Preprint*, arXiv:2304.02623.

Bowen Tan, Zichao Yang, Maruan Al-Shedivat, Eric Xing, and Zhiting Hu. 2021. Progressive generation of long text with pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4313–4324, Online. Association for Computational Linguistics.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Yidong Wang, Qi Guo, Wenjin Yao, Hongbo Zhang, Xin Zhang, Zhen Wu, Meishan Zhang, Xinyu Dai, Min Zhang, Qingsong Wen, Wei Ye, Shikun Zhang, and Yue Zhang. 2024. Autosurvey: Large language models can automatically write surveys. *Preprint*, arXiv:2406.10252.

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. 2024. A survey on large language models for recommendation. *World Wide Web*, 27(5):60.

Peng Xu, Wei Ping, Xianchao Wu, Chejian Xu, Zihan Liu, Mohammad Shoeybi, and Bryan Catanzaro. 2024. Chatqa 2: Bridging the gap to proprietary llms in long context and rag capabilities. *Preprint*, arXiv:2407.14482.

Erguang Yang, Mingtong Liu, Deyi Xiong, Yujie Zhang, Yufeng Chen, and Jinan Xu. 2022. Long text generation with topic-aware discrete latent variable model. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8100–8107, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Kevin Yang, Dan Klein, Nanyun Peng, and Yuandong Tian. 2023. DOC: Improving long story coherence with detailed outline control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3378–3465, Toronto, Canada. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Mingjia Yin, Hao Wang, Wei Guo, Yong Liu, Suojuan Zhang, Sirui Zhao, Defu Lian, and Enhong Chen. 2024. Dataset regeneration for sequential recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3954–3965.

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

Chengyue Yu, Lei Zang, Jiaotuan Wang, Chenyi Zhuang, and Jinjie Gu. 2024. CharPoet: A Chinese classical poetry generation system based on token-free LLM. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 315–325, Bangkok, Thailand. Association for Computational Linguistics.

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 [cs].

A Dataset Details

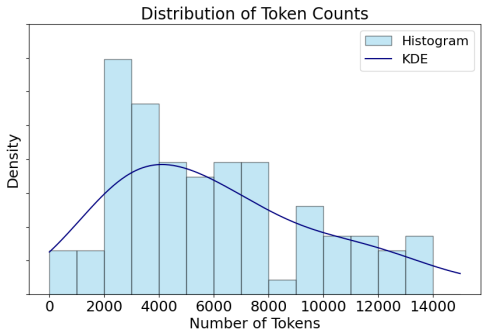

Figure 3: The distribution of the length of FreshWiki-2024

We adopted the data collection methodology outlined in FreshWiki to construct our dataset. Specifically, we systematically crawled the top 100 most frequently edited topics monthly from January to December 2024, retaining only articles rated as Class B or higher by the ORES quality assessment system[7]. To streamline processing and focus our analysis, we removed all tabular data and multimodal content, preserving only plain text following the same approach as STORM.

From this curated collection, we randomly selected 100 topics for model evaluation. As shown in Figure 3, article lengths followed a bimodal distribution: approximately 70% of articles ranged between 2,000 and 8,000 tokens, while we intentionally retained articles exceeding 15,000 tokens to evaluate the model's performance or processing intricate topic matter.

To ensure thematic diversity, We used GPT-4o to categorize 100 selected topics into nine main groups based on Wikipedia's official classification standards. Figure 4 illustrates the resulting category distribution, demonstrating broad coverage and relatively balanced representation across subject areas.

B Evaluation Details

B.1 Limitations for NER-based outline evaluation methods

STORM's NER-based evaluation metrics, namely Entity Recall and Soft Entity Recall, exhibit two fundamental limitations. First, conventional NER models are not specifically designed for outline evaluation. In particular, key section titles (e.g.,

[7]https://www.mediawiki.org/wiki/ORES

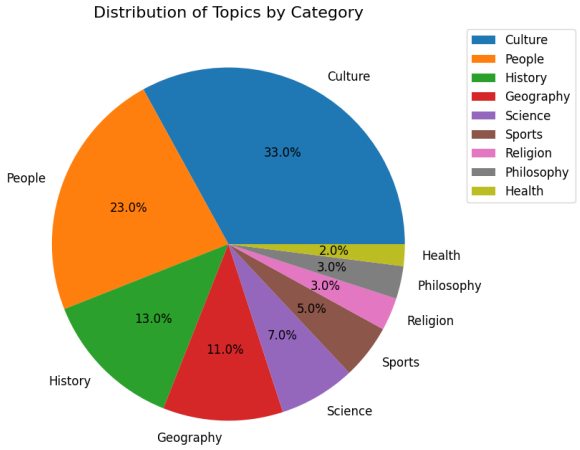

Figure 4: The distribution of classification of Freshwiki-2024

"Background") in outlines often do not correspond to well-defined named entities. As a result, extracting entities from golden outlines can be unreliable, and in extreme cases, no identifiable entities may be present at all. Second, the soft entity recall metric suffers from abnormal scaling behavior due to the lack of a normalization mechanism. This absence of normalization, coupled with the inherent limitations of entity recognition, can lead to inflated scores that misrepresent the actual quality of an outline.

B.2 Article Quality Evaluation

We utilize the Qwen-Turbo model as the backbone for FactScore, which decomposes atomic facts and assesses their accuracy. FactScore can be regarded as a measure of the factual accuracy of generated text. However, comparing FactScore across different texts can be challenging when the generated content varies significantly in length. To address this issue, we utilize FactAlign, which assumes a predefined number of claims, denoted as K, for each article. The recall is then computed as the ratio of generated claims to K, allowing for a more balanced evaluation. Based on this, we calculate the $F1@K$ score, where we set $K = 300$ in our experiments.

Meanwhile, we follow existing work (Shao et al., 2024; Dong et al., 2025b) to utilize LLM for automatic evaluation. We observe that due to the context length limitations of `Prometheus-7b-v1.0`, STORM truncates all generated articles to a limited length of 2,000 words, which significantly impacts the fairness of the evaluation. Thanks to the extended context length of `Prometheus-7b-v2.0`, we were able to process articles up to 10,000 words in our experiments, leading to a more comprehensive assessment. The detailed prompts for LLM Evaluation are shown in Table 6. Table 5 shows the Pearson correlation coefficient data between human evaluation and LLM evaluation. Our preliminary correlation analysis of this evaluation model revealed relatively low correlation for the Relevance metric, possibly due to ambiguous topics and the LLM's tendency to be overly lenient. For instance, "vultures 1" (an album) was considered relevant even when "vultures" (the animal) was the primary topic. For a more comprehensive correlation analysis of this evaluation approach, refer to STORM (Shao et al., 2024) and Prometheus (Kim et al., 2024). It is important to note that the evaluation of long articles remains a challenging problem, and existing evaluation models are not yet perfect. We consider this a direction for future research.

	Pearson Correlation
Interst Level	0.58
Organization	0.51
Relevance	0.31
Coverage	0.47

Table 5: Pearson correlation between average human evaluation scores and LLM-based evaluation scores on genreated article quality (n=20)

Additionally, we utilized `text-embedding-3-small` to generate embeddings for the info diversity metric, following the implementation of CoSTORM.

C Full Prompt

As discussed in §2, we divided our RAPID framework into three modules and listed the prompts used for each. The prompts we use for §§3.2 are listed in Figure 6, 7 while §§3.3 in Figure 8, 9, 10, 11, 12 and §§3.4 in Figure 13, 14.

D Efficiency and Usage Analysis

RAPID integrates an attribute-constrained information-seeking module, which is designed to enhance efficiency while maintaining comprehensiveness. To evaluate its efficiency and usability compared to other baselines, we collected the average time and API usage for generating 100 topics. For methods involving parallel section generation, we set the maximum parallelism to

Criteria Description	**Interest Level**: How engaging and thought-provoking is the article?
Score 1 Description	Not engaging at all; no attempt to capture the reader's attention.
Score 2 Description	Fairly engaging with a basic narrative but lacking depth.
Score 3 Description	Moderately engaging with several interesting points.
Score 4 Description	Quite engaging with a well-structured narrative and noteworthy points that frequently capture and retain attention.
Score 5 Description	Exceptionally engaging throughout, with a compelling narrative that consistently stimulates interest.
Criteria Description	**Coherence and Organization**: Is the article well-organized and logically structured?
Score 1 Description	Disorganized; lacks logical structure and coherence.
Score 2 Description	Fairly organized; a basic structure is present but not consistently followed.
Score 3 Description	Organized; a clear structure is mostly followed with some lapses in coherence.
Score 4 Description	Good organization; a clear structure with minor lapses in coherence.
Score 5 Description	Excellently organized; the article is logically structured with seamless transitions and a clear argument.
Criteria Description	**Relevance and Focus**: Does the article stay on topic and maintain a clear focus?
Score 1 Description	Off-topic; the content does not align with the headline or core subject.
Score 2 Description	Somewhat on topic but with several digressions; the core subject is evident but not consistently adhered to.
Score 3 Description	Generally on topic, despite a few unrelated details.
Score 4 Description	Mostly on topic and focused; the narrative has a consistent relevance to the core subject with infrequent digressions.
Score 5 Description	Exceptionally focused and entirely on topic; the article is tightly centered on the subject, with every piece of information contributing to a comprehensive understanding of the topic.
Criteria Description	**Broad Coverage**: Does the article provide an in-depth exploration of the topic and have good coverage?
Score 1 Description	Severely lacking; offers little to no coverage of the topic's primary aspects, resulting in a very narrow perspective.
Score 2 Description	Partial coverage; includes some of the topic's main aspects but misses others, resulting in an incomplete portrayal.
Score 3 Description	Acceptable breadth; covers most main aspects, though it may stray into minor unnecessary details or overlook some relevant points.
Score 4 Description	Good coverage; achieves broad coverage of the topic, hitting on all major points with minimal extraneous information.
Score 5 Description	Exemplary in breadth; delivers outstanding coverage, thoroughly detailing all crucial aspects of the topic without including irrelevant information.

Table 6: Scoring Rubrics for both Human and LLM Evaluation

	Calls(/it)	Tokens(k/it)	Time(s/it)
RAG	2	6.37	32.12
ORAG	14.25	24.13	109.83
STORM	88.06	60.71	163.22
Co-STORM	70.30	50.52	154.14
RAPID	**31.04**	**43.62**	**127.19**

Table 7: Results of pipeline efficiency evaluation. We evaluated the average API calls, token consumption, and the total duration time of generating an article.

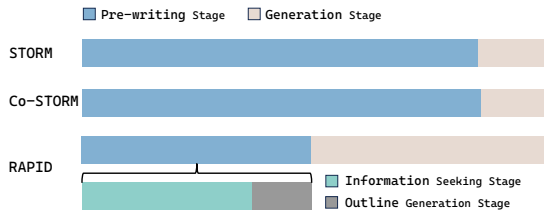

Figure 5: The distribution of time consumed in each stage of the pipelines.

3. As shown in the table, although the first three simple baselines consume less time and resources, they struggle to generate reliable articles. In contrast, RAPID not only produces high-quality articles but also significantly reduces API usage and time consumption compared to STORM and Co-STORM. Additionally, we analyzed the proportion of time consumed at each stage by STORM, Co-STORM, and RAPID, as illustrated in Figure 5. The results indicate that both STORM and Co-STORM predominantly allocate a substantial amount of time to the pre-writing stage, leading to significant time consumption. This is largely due to their reliance on an agent-based dialogue mechanism during the pre-writing phase, which is inherently difficult to optimize further. In contrast, RAPID drastically reduces the time spent in the pre-writing stage through its parallelized information collection approach, enabling more efficient resource allocation across all stages.

E Case Study

We utilized RAPID to generate an article on the topic of AlphaFold. The complete outline and writing plan are illustrated in Figures 15 and 16, respectively. Due to space limitations, only a small portion of the final generated article is presented in Figure 17. This case demonstrates that RAPID is

capable of effectively generating a coherent writing plan and producing a comprehensive and consistent long-form article.

Topic Summarization

You are an expert in utilizing search engines effectively. You are currently compiling information for a wiki page based on a given topic. Now that you have obtained content returned by search engines regarding this topic, please analyze whether there are any ambiguities or multiple concepts associated with it. If the topic is clear, generate a brief introduction based on the search engine content to clarify the concept for subsequent writing, ensuring that the introduction remains within three sentences.

Topic you are discussing about: <topic>

Gathered information from search engines: <search results>

Now give your response. Make sure that only the introduction is outputted. Do not repeat the input prompt.

Figure 6: Prompt used for Topic Summarization

RAG Outline Generation

Write an outline for a Wikipedia page.
Here is the format of your writing:
1. Use "#" Title" to indicate section title, "##" Title" to indicate subsection title, "###" Title" to indicate subsubsection title, and so on.
2. Do not include other information.
3. Do not include topic name itself in the outline.

The topic you want to write: <topic>

Brief intro of the topic: <brief info>

Outlines of similar topics: <similar topic outline>

Write the outline of the topic:

Figure 7: Prompt used for RAG Outline Generation

Attributes Extraction

Generate attributes for a specified topic with its outline. The generated attributes should summarize all information needed to write the wiki page for this topic. Please avoid creating complex attributes; ensure that each attribute represents a distinct and indivisible aspect.
Here is the format of your response:
1. Generate attributes, each on a new line, ensuring no additional tags or formatting are included.
2. Do not include other information.
3. Do not include topic name itself in the attribute list.

Topic: <topic>

Outline: <outline>

Attributes:

Figure 8: Prompt used for Attributes Extraction

Attributes to Queries

You want to search the info of attributes of the topic using Google search. What do you type in the search box? Write the queries you will use in the following format:
- query 1
- query 2
- ...
- query n

Topic you are discussing about: <topic>

The attributes of the topic: <attributes>

Now give your response. Make sure that only queries are output. Do not repeat the input prompt

Figure 9: Prompt used for Attributes to Queries

Queries Merging

I want you to act as a researcher gathering information to compose a wiki article on a specific topic. You are now presented with a topic and a list of queries designed to gather information for the topic. Your task is to modify or enhance the query list based on the relevant topics and their queries. Ensure that the final queries comprehensively encompass information beneficial for writing about the topic and are suitable for use in Google searches. Do not repeat the input prompt. Here is the format of your response:
- query 1
- query 2
- ...
- query n

The topic you are discussing about: <topic>

The queries of the topic: <queries>

The similar topics with their queries: <similar topics with queries>

The final response of the queries:

Figure 10: Prompt used for Queries Merging

Operation Generation

You are improving an outline for a wiki page. Now I will give you a draft outline and some titles of the searched results. You can do three operations:
add section
delete section
do nothing

Please list the operations you need to do:
-[add section] : section_title
-[delete section] : section_title
If nothing is needed to do, please just generate:
-[do nothing]

Directly write the operations and do not include any other information.

The topic you want to write: <topic>

The draft outline: <outline>

Titles of the searched results: <titles>

Please generate the operations:

Figure 11: Prompt used for Operation Generation

Outline Refinement

You are improving an outline for a wiki page. Now I will give you a draft outline and some operations like:
-[add section]
-[delete section]
-[do nothing]
Please proceed with the operations for the outline and then refine the overall outline. Directly write the refined outline and do not include any other information.

The topic you want to write: <topic>

The draft outline: <outline>

The operations: <operations>

Please generate the refined outline:

Figure 12: Prompt used for Outline Refinement

Plan Generation

You are an experienced wiki writer. I will provide you with a topic with its outline to write. I want you to generate a writing plan for this outline to improve the coherence of the article. The plan defines which sections is needed to be generated before the current section. Try to choose the sections that can help improve the coherence and fluency of the current section. For example, sections like 'Background' don't need extra information while sections like 'Introduction' or 'Conclusion' need all other sections. Please just generate the plan for the first level sections and make sure that the plan is in a valid topological order. If no extra information is needed, generate "None". All the needed sections are connected by '<-' and make sure that they are all from the first level sections of outline. Just output the plan and do not explain.

Here is an example: <example>

Topic: <topic>

Outline: <outline>

Generate the plan of the given topic and outline(do not repeat the outline):

Figure 13: Prompt used for Plan Generation

Section Writing

Write a Wikipedia section based on the collected information. Here is the format of your writing:
1. Use "#" Title" to indicate section title, "##" Title" to indicate subsection title, "###" Title" to indicate subsubsection title, and so on.
2. Use [1], [2], ..., [n] in line (for example, "The capital of the United States is Washington, D.C.[1][3]."). You DO NOT need to include a References or Sources section to list the sources at the end.
The collected information: <collected info>

The topic of the page: <topic>

The other sections of the page: <other sections>

The section you need to write: <section title>

The outline of the section: <section outline>

Write the section with proper inline citations (Start your writing with # section title. Don't include the page title or try to write other sections):

Figure 14: Prompt used for Section Writing

AlphaFold

Background
Development
Origin of the project
Collaboration with scientific communities
Algorithm
Neural network architecture
Training datasets
AlphaFold 1
AlphaFold 2
Updates and enhancements in AlphaFold 3
AlphaFold 3 Pairformer and advancements
Protein Structure Predictions
Methodology
Accuracy and reliability
Comparison with experimental methods
Competitions and Benchmarks
CASP12
CASP13
CASP14
CASP15
Applications
Drug discovery
Enzyme engineering
Understanding diseases
Environmental research
Synthetic biology
Role in antibody characterization
Use of AlphaFold in predicting genetic mutation impacts
Database
AlphaFold Protein Structure Database
Accessibility and usage
Global impact on research
Source Code and Open Access
Release of AlphaFold source code
Community contributions and forks
Limitations
Known issues
Challenges in complex protein assemblies
Limitations with membrane proteins and disordered regions
Limitations in predicting intrinsically disordered regions
Reception and Impact
Scientific community
General public
Ethical considerations
Future Directions
Enhancements in accuracy
Integration with other biological tools
Expansion to other molecular complexes

Figure 15: The complete outline of AlphaFold generated by RAPID.

AlphaFold

Background: Development
Development: Algorithm,
Algorithm: Protein Structure Predictions, Competitions and Benchmarks, Limitations
Protein Structure Predictions: Applications, Limitations
Competitions and Benchmarks: None
Applications: Database,Reception and Impact
Database: Source Code and Open Access
Source Code and Open Access: None
Limitations: Reception and Impact
Reception and Impact: Future Directions
Future Directions: None

Figure 16: The writing plan for AlphaFold generated by RAPID. Each section is a dependency for the sections listed as its values; "None" indicates that the section does not serve as a dependency for any other section.

AlphaFold

Development
Origin of the Project
The development of AlphaFold emerged from a long-standing challenge in the field of structural biology: accurately predicting protein structures based on amino acid sequences, a problem often referred to as the "protein-folding problem." This issue had persisted as one of the most significant puzzles in biology for over five decades, with limited progress despite numerous scientific efforts[2]. DeepMind, a subsidiary of Alphabet Inc., initiated the AlphaFold project with the goal of leveraging advances in artificial intelligence, particularly deep learning, to address this challenge. The project culminated in the release of AlphaFold 2 in 2020, a system that achieved near-experimental accuracy in many cases and fundamentally reshaped the landscape of protein structure prediction[3].

Collaboration with Scientific Communities
The development and adoption of AlphaFold have been significantly bolstered by its integration with and support from scientific communities. Following its launch, AlphaFold quickly became a focal point of discussion and collaboration within the structural biology field. For instance, researchers like Pedro Beltrao from ETH and EMBL-EBI initiated joint studies based on results shared on social media platforms, turning the excitement surrounding AlphaFold into a collective research effort. This bottom-up approach exemplified the enthusiasm and openness fostered by AlphaFold's innovations, allowing scientists from various institutions to collaborate and build upon shared findings[4]. Moreover, AlphaFold embraced the principles of open science by releasing its predictions and source code, alongside the establishment of an open-access protein structure database containing over 200 million protein structures. This democratized access to cutting-edge resources and enabled researchers globally to integrate AlphaFold's predictions into their work, regardless of institutional funding or geographic location. By adhering to transparency, accessibility, and collaboration, AlphaFold not only advanced the field of protein structure prediction but also inspired open science initiatives in other disciplines[2].

Algorithm
AlphaFold 1
The first iteration of AlphaFold debuted in the Critical Assessment of Structure Prediction (CASP13) competition, marking a significant milestone in protein structure prediction. While AlphaFold 1 relied on traditional homology modeling approaches combined with deep learning techniques, its ability to predict protein structures with reasonable accuracy laid the groundwork for future advancements. This version introduced key innovations in handling sequence alignments and template-based predictions, setting the stage for subsequent breakthroughs[5].

AlphaFold 2
AlphaFold 2 represented a transformative leap in protein structure prediction. Unveiled during CASP14, it utilized a completely redesigned architecture to achieve near-experimental accuracy in many cases. Unlike its predecessor, AlphaFold 2 did not rely solely on homology modeling or templates but could predict structures for previously unknown protein folds[6][5]. The model introduced innovations such as the incorporation of attention mechanisms and pairwise geometric features, enabling it to accurately model spatial relationships between amino acids. These advances allowed AlphaFold 2 to excel at decoupling the training and inference tasks, a unique approach that optimized the system for both learning from data and making predictions[7][6].

Figure 17: A segment of the AlphaFold article generated by RAPID.